Building dynamic spatial environmental models

Het maken van dynamische ruimtelijke landschapsmodellen

(met een samenvatting in het Nederlands)

Proefschrift

ter verkrijging van de graad van doctor aan
de Universiteit Utrecht op gezag van de Rector Magnificus,
Prof. dr. W.H. Gispen, ingevolge het besluit van het College
voor Promoties in het openbaar te verdedigen op 8 november
2002 des ochtends te 10.30 uur

door
Derek Karssenberg
geboren op 25 november 1968, te Utrecht

Promotores:

Prof. Dr. P.A. Burrough	Utrecht University
Prof. Dr. M.F.P. Bierkens	Utrecht University

Co-promotor:

Dr. W.P.A. van Deursen	Carthago Consultancy

Building dynamic spatial environmental models

Nederlandse Geografische Studies / Netherlands Geographical Studies

ISSN 0169-4839

Netherlands Geographical Studies 305

Building dynamic spatial environmental models

Derek Karssenberg

Utrecht 2002

Koninklijk Nederlands Aardrijkskundig Genootschap/
Faculteit Ruimtelijke Wetenschappen, Universiteit Utrecht

This publication is identical to a dissertation submitted for the title of Doctor at Utrecht University, the Netherlands. The public defence of this thesis took place on November 8, 2002.

Promotores:

Prof. Dr. P.A. Burrough	Utrecht University
Prof. Dr. M.F.P. Bierkens	Utrecht University

Co-promotor:

Dr. W.P.A. van Deursen	Carthago Consultancy

Examination committee:

Prof. Dr. P.L. de Boer	Utrecht University
Prof. Dr. D.E. Walling	University of Exeter
Prof. Dr. Ir. A.K. Brecht	Wageningen University
Drs. M. de Bakker	Van Hall Institute
Dr. G.B.M. Heuvelink	University of Amsterdam

ISBN 90-6809-341-X

Printed in the Netherlands by Labor Grafimedia b.v. - Utrecht

CONTENTS

List of figures

List of tables

1 SCOPE, PROBLEM DEFINITION AND RESEARCH QUESTIONS

1.1 Introduction

A model of the environment, including landscapes, is a representation or imitation of complex natural phenomena that can be discerned by human cognitive processes. Models of the landscape are almost always representations in miniature even though the representation is physical (an analogue model) or in mathematical equations. Since the landscape or environment is a complex system under continuous change, most mathematical models can only be run on a computer. This thesis deals with the type of mathematical computer models that will be referred to as dynamic spatial environmental models. The word 'spatial' refers to the geographic domain which they represent, which is the two- or three-dimensional space, while 'dynamic' refers to models simulating changes through time using rules of cause and effect. A more exact definition will be given in Section 1.2. Examples of dynamic spatial environmental models are computer models simulating the flow of water in an area, the spreading of a species over a continent, or the transport of pollutants through the air.

Dynamic spatial environmental models have a scientific value mainly because they can be used to improve the understanding of environmental processes, for instance to test hypotheses regarding the driving forces of changes in the environment. These models also provide a means to communicate scientific knowledge. Finally, dynamic spatial models are all-important for environmental planning and management, since they can be used to make predictions for future behaviour of an environmental system, or to evaluate the impact of changes in the environment made by people or other organisms.

As will be explained in the following sections, most dynamic spatial environmental models cannot be regarded as all purpose tools that can be applied off-the-shelf. This is because each case study has specific conditions to which the model needs to be tailored. As will be explained in the following sections, these conditions or constraints include among others the aim of modelling, the properties and processes of the study site, the available field data and the computer technology that can be used. Model building, which is the subject of this thesis, involves finding the optimal model for a specific case study given these conditions. In this thesis, the term 'model building' refers to the identification of all equations in a model and their implementation in a software program, but also the identification of appropriate inputs and parameters needed in the model.

Model building is a difficult issue since it involves the diverse disciplines of computer technology, mathematics, and environmental science, and thus environmental experts are needed who are trained in all these disciplines. Four central issues related to model building are treated in this thesis. These are: 1) programming and tools to program the model, 2) estimation of inputs and parameters of a model from field data by upscaling, 3) estimation of inputs and parameters from field data by inverse modelling, and 4) training researchers in model building.

Section 1.2 is a short introduction to dynamic spatial modelling providing definitions of the most important terms used in this thesis. Section 1.3 defines the problem definition,

and section 1.4 gives the research questions. Section 1.5 provides an outline of the remaining chapters in the thesis.

1.2 Introduction to dynamic spatial environmental models

Environmental models are considered here as a representation of a part of the landscape, including things and processes above, at or below the land or water surface. The entities represented may be objects or continuous spatial fields as studied by natural sciences such as biology, ecology, physical geography, geology, or meteorology. Environmental models can be physical or mathematical. *Physical* and *analogue models* represent an environmental system with a miniature version of that system in a laboratory. An example of an analogue model is a tank filled with (flowing) water and sediment to mimic sedimentation and erosion on a continental shelf (e.g., Hasbargen and Paola, 2000). *Mathematical models* use mathematical equations as a model of the environment. Besides other types of mathematical models, such as statistical models, *dynamic models* are widely used in the environmental sciences. The property of a model that makes it dynamic is that it is run forward in time, using rules of cause and effect to simulate temporal changes in the landscape. For this reason, some people refer to dynamic models as forward models. Although dynamic environmental models are often spatial models, in the sense that they represent spatial entities in a landscape, for instance a soil layer, it is easier to restrict ourselves first to the non-spatial, *point model*. For this case, the concept of *dynamic modelling* can be represented by the following equation, which is illustrated in Figure 1.1. Similar descriptions can be found in Beck *et al.* (1993), Gurney and Nisbet (1998), and Van Deursen (1995):

$$z(t+1) = \mathrm{f}\big(z(t), i(t), t\big) \quad \text{for each } t \qquad (1)$$

In this equation, a certain property, or attribute, of the landscape is represented by the non-spatial state *variable z*, which can be a continuous variable, for instance temperature, or a classified variable, for instance vegetation class. It could also represent an attribute of an individual, for instance the height of a tree. This variable z has a value at each moment in time t, and the value of z at that moment is written as $z(t)$, while $z(t+1)$ means the value of z at a certain moment later in time. Although z changes in a continuous way through time, which can be represented by a set of differential equations, a discrete representation of time is used here. The letter f represents a *functional* with associated *parameters*, operating on the variables inside the brackets. It can be either an update rule, explicitly specifying the change of the state variable over the time slice $(t, t+1)$, for instance a rule based function such as cellular automata (e.g., Toffoli, 1989), a probabilistic function, or alternatively a derivative of a differential equation describing the change of the state variables as a continuous function (c.f., Gurney and Nisbet, 1998). Equation (1) shows that for each moment in time, the value of the attribute at that moment ($z(t)$) is used to calculate that value of the attribute at a later moment in time. This change in z over the period $(t, t+1)$ is represented by the functional f. The second term $i(t)$ can be zero for all

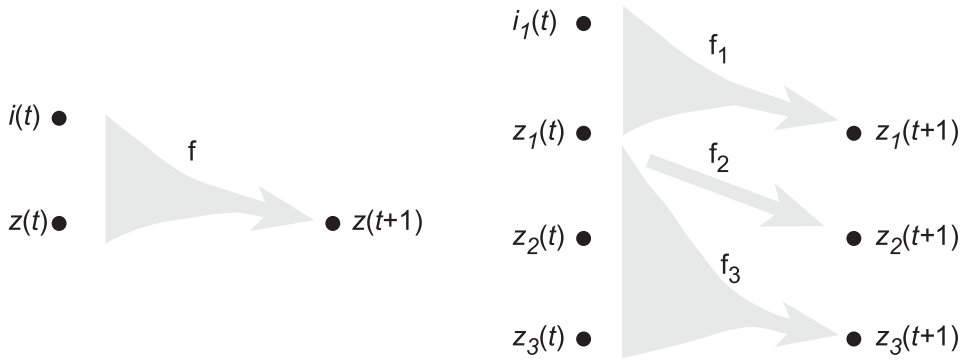


Figure 1.1. Dynamic non spatial model; $i_j(t)$, inputs; $z_k(t)$, state variables; f_k, functionals. Left, one input and one state variable; right, multiple inputs (only one shown) and state variables (three shown).

time steps, representing a dynamic model which is not affected from outside, a closed system. But in many cases, the system represented by a dynamic model has external *inputs*, sometimes called disturbances. Think for instance of rain falling on the ground surface in a model simulating infiltration of water into the soil, or addition of nutrients from agriculture in a model simulating a lake ecosystem. Such inputs are represented in equation (1) by $i(t)$, which is, just like z, defined for each moment in time t. Boundary conditions needed in a model are regarded here as an input $i(t)$ too. The functional f operates on this input and the state variable, as shown in Figure 1.1. Note that some models derive $z(t+1)$ from an input only, which means that equation (1) reduces to $z(t+1) = f(i(t))$. These models are also regarded as dynamic models. Finally, the functional f uses the time t to calculate $z(t+1)$, which is needed when the processes in the landscape change with time, for instance as a result of climate change.

Equation (1) represents the simple case where one state variable z is used, ignoring interaction between different components or processes in an environment, such as predator-prey relations, or deposition of sediment controlled by water level, flow speed, or sediment type. Models that simulate interaction between different components of an environment need to include a set of state variables, as illustrated in Figure 1.1 (right), which can be represented by:

$$z_k(t+1) = f_k\left(z_k(t), k = 1..m; i_j(t), j = 1..n; t\right) \quad \text{for each } t \qquad (2)$$

For each state variable z_k, where k represents one of the 1 to m state variables involved, its value at $t+1$ results from a functional f_k on all (or a part of) the state variables $z_k(t)$, $k = 1..m$. In addition, each variable z_k can be determined by a set of $j = 1..n$ inputs i_j. While f in equation (1) was a relatively simple functional, f_k in equation 2 can be rather complex, representing a complex set of interactions between state variables and inputs, which is called the *model structure*.

Although the components of some environmental systems can be described by a non-spatial, or one dimensional dynamic model, many environmental processes include important spatial interactions. For representing these processes, the dynamic model becomes a *spatial model*, and needs to consider the environment as a two or three

dimensional spatial system, and the variables need to be represented in a two or three dimensional domain D, while the variables have a value at all locations or a part of the locations in this domain. Such spatial variables are represented by $z_k(\mathbf{s},t)$, where $\mathbf{s}$ is the spatial index of z_k, representing the subset of D where z_k exists. In the case that z_k represents an attribute of a continuous field, for instance the topographical elevation, z_k has a value for all locations in D, and $\mathbf{s}$ is an index which varies continuously. The value of the variable z_k at t+1 is an integral over the spatial domain D of a functional f_k on the state variables, inputs, space and time, as illustrated in Figure 1.2:

$$z_k(\mathbf{s}_0, t+1) = \int_D f_k\left(z_k(\mathbf{s},t), k = 1..m; i_j(\mathbf{s},t), j = 1..n; \mathbf{s}, t\right) d\mathbf{s} \quad \text{for each } t \qquad (3)$$

When z_k represents an attribute of one or more objects, for instance the age of a set of birds, $\mathbf{s}$ becomes a set of locations $\mathbf{s}_i$ in D, representing the locations of the birds, and (3) becomes a summation over these locations instead of an integral. Similar as in the non spatial case, each functional f_k operates on all (or part of) the variables z_k, k = 1..m. For each variable, the functional operates on all values $z_k(\mathbf{s},t)$ on the spatial index $\mathbf{s}$, and also on the location and the moment in time, represented by the additional inputs $\mathbf{s}$ and t. The same holds for the inputs i_j. Note that in the spatial case, many parameters used in the functionals will have values changing in the spatial dimension. To illustrate the abstract equation 3, think of a dynamic model representing a forest ecosystem. Spatially continuous attributes such as soil moisture content, or shrub vegetation biomass, will be modelled with continuous field state variables. Individuals, such as animals, will be represented by state variables representing objects, which move in space. The dynamic model would include many spatial functionals representing the behaviour of such a system, where the state variables related to properties of the animals change as a functional of continuous fields, such as available biomass, while these continuous fields could change as a functional of other continuous fields, or the location and behaviour of the animals, for instance by consumption of vegetation.

In addition to the classification of dynamic models in non-spatial and spatial models, dynamic models can be either deterministic or stochastic. A *deterministic model* has state variables which have a single value for each location in space and moment in time. A *stochastic model* deals with state variables which in the non-spatial case are random variables, having a certain probability distribution, or in the spatial case, random fields. A dynamic model becomes a stochastic model when its inputs i_j or parameters are stochastic (c.f., Heuvelink, 1998), or when the functional f_k involves a probabilistic rule.

So far, we have mainly looked at how an environmental system works, and how it can be represented by a model. In many cases, and always in applied research, studying this system is not the main aim of modelling, but the model is merely an instrument to predict a specified set of properties of the system. If we look at the example of the forest ecosystem, the main aim would be to predict for instance the number of deer or the average biomass production in a certain area, as well as understanding the system as a whole. In these cases, a model is regarded as a system producing a certain number outputs in which the interest lies. This is shown in Figure 1.3, which can be regarded as a summary of what is stated in this section: a model has external inputs $i_{1..k}$, internal state

variables $z_{1..m}$, functionals $f_{1..m}$ with associated parameters **p**, while it generates a certain number of *outputs*, which are some of the state variables in the model.

Although dynamic models which are restricted to non-spatial, deterministic simulation can sometimes be solved analytically, spatial and/or stochastic dynamic models include interactions that in most cases are too difficult to solve without a numerical solution scheme. So, in most cases, dynamic spatial models are *numerical models*, which need to be programmed and run on a computer.

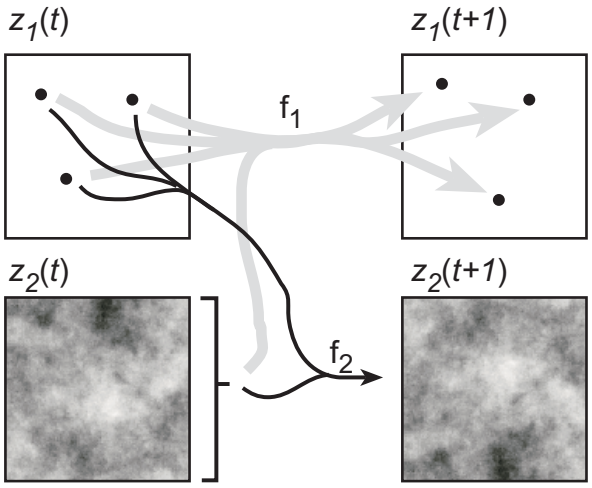


Figure 1.2. Dynamic model in two dimensional space; $z_k(t)$, state variables; f_k, functionals. Inputs $i_j(t)$ not shown, only two state variables shown, $z_1(t)$ a state variable representing objects, $z_2(t)$ a state variable representing a continuous field.

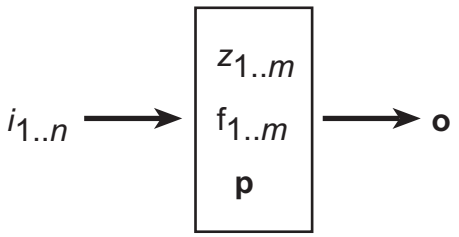


Figure 1.3. Dynamic spatial model as an open system with inputs $i_{1..n}$, state variables $z_{1..m}$, functionals $f_{1..m}$, parameters **p** and output variables **o**.

1.3 Problem definition

1.3.1 Factors involved in model building

The aim of model building is to find the optimal representation of environmental processes in the numerical equations (and parameters) of a computer program. The qualitative term optimal is used here, since models can only be judged in the context of many factors involved in a modelling study (Figure 1.4), and some of these factors can only be interpreted in qualitative terms. Since problems related to model building are all part of one or more factors, a short discussion of the main factors is given below.

Theory of natural sciences. The model structure needs to be defined on the basis of the knowledge of the environmental system which is to be modelled, and is a synthesis of laws known, or hypothesised about, in the natural sciences. There is a large range of approaches for achieving a synthesis, with two extremes. On one side of this range is the approach that aims at combining laws used, or closely related to, those known in more generic sciences such as physics or chemistry. For instance, the kinematic wave equations for simulating overland flow can be cut down to general physical laws. I will refer here to these kind of models as *process driven models*, because their equations are derived from lower level laws which can be assumed to be valid under all known circumstances. The term physically based models, which is often used, is not used here for these kind of models, since lower level laws do not always need to be physical laws. Being a synthesis of lower level laws, the model structure of a process driven model is a highly complex set of interactions, represented by a large number of laws, which mostly need to be given on a small spatial and temporal resolution. Consequently, process driven models typically include a large number of equations and parameters, using state variables and inputs represented at a high resolution (Figure 1.5). Examples of the process driven approach to modelling in hydrology are distributed watershed models such as the SHE model (Abbott *et al.*, 1986) and the unifying modelling framework for watershed dynamics described by Reggiani *et al.* (1998).

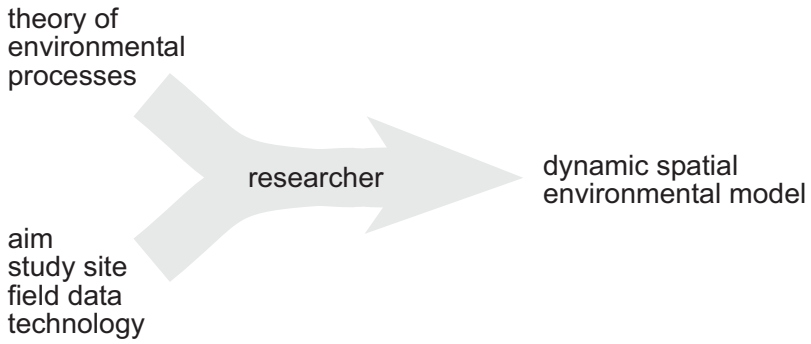


Figure 1.4. Factors in dynamic spatial environmental model building. The theory of environmental processes is given separate from the other factors, since it is a general factor, while the other factors are site specific.

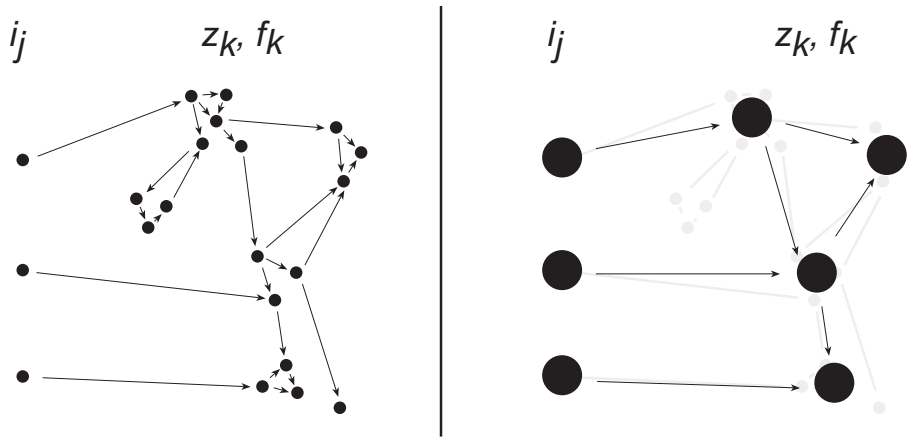


Figure 1.5. Process driven (left) and data driven (right) representation of an environmental system; i_j, inputs; z_k, state variables; lines represent functionals f_k; small dots represent high resolution, large dots represent low resolution.

An equally valid approach on the other side of the range of approaches is to combine laws which are valid not because they can be derived from lower level laws, although this might still be the case, but mainly because they have, or can be, shown to be valid for describing environmental processes. These laws can be derived from process driven models, or are empirically derived, using field data of environmental attributes. Their scientific value lies not in their derivation from lower level laws, but mainly in the fact that they are set up according to rules of logic combined with knowledge of environmental processes. For this reason, these laws are sometimes, but not always, only valid under certain conditions defined by the properties and processes of the study site under consideration. I will refer to these models as *data driven models*, since their laws are mostly derived from environmental data. These models are sometimes refered to as empirically based models, in contrast to physically based models, but the term empirically based models is not used here, mainly because laws in physics or chemistry are also empirically derived. Since data driven models are a synthesis of higher level laws each incorporating a large amount of system behaviour, these models typically include a small number of equations and parameters, valid at a lower spatial and temporal resolution (Figure 1.5). An example of a data driven model is a model simulating pollutant transport in a large catchment built upon laws derived from field data at the scale of modelling and knowledge of environmental processes at that scale (e.g., de Wit and Pebesma, 2001). Another example of model building resulting in data driven models is the downward approach to hydrological model development (Klemes, 1983; Jothityangkoon *et al.*, 2001).

Aim of modelling. The representation of processes in a model also depends on the aim of modelling (Jørgensen, 1988). A large group of models can be regarded as being built mainly to encapsulate all existing knowledge of the processes being modelled. These models are typically process driven models, with a complex model structure in the sense that a large number of equations is coupled to represent the whole environmental system. In many cases, these models are difficult to run with field data as input, since they need a large number of input data to parameterise all the model equations, which is

not always available. As a result, these models have mainly a scientific value, meant to improve understanding of environmental processes and to communicate this understanding. The same holds for simplified models which focus on a small subset of processes in a system, using simplified equations. Such models are the key to develop a general theory, or help us understand the behaviour of more complex models (c.f., Gurney and Nisbet, 1998; Casti, 1998), and they are mostly run without in situ field data. Another group of models is built mainly aiming at making predictions for a specific purpose, for instance as part of a decision support system (Sprague and Watson, 1982). The structure of these models needs to be tailored to the outputs for which predictions need to be made, the spatial and temporal resolution for which predictions are needed, and the prediction precision which satisfies the aim of the study, under the constraint of the model input data available. In many cases, the models used as management tools are data driven models, since data driven models are by definition tailored to the available input data and the outputs required. But in some cases, process driven models can be used too, when their inputs and outputs fit the study under consideration.

Study site and field data. A model which is meant to represent environmental processes at a specific study site, needs to embody processes that are important at the study site. The set of processes that is dominant will be different between different study sites, which is one of the reasons why in many cases one and the same model cannot be applied to a number of study sites (Beven, 2000). In addition, when a data driven model is used, the model structure will by definition be dependent on the properties of the field data available, where properties refers to the amount, spatial and temporal resolution, and precision of field data. The field data are used to find correct values for the inputs and parameters in the model. When building a model, a balance needs to be found between the properties of the field data and the complexity of the model structure (e.g., Jørgensen, 1988; Donnelly-Makowecki and Moore, 1999; van der Perk, 1997; Beven, 2000; de Wit and Pebesma, 2001). This is another important reason why a model cannot be regarded as a fixed thing, with generic application. Instead, it needs to be tailored to the field data available.

The researcher. Although the theory and field data are the main factors determining the model structure, the somewhat subjective role of the researcher cannot be neglected, since it is the researcher that needs to synthesise theory and field data, for a given aim and study site. As noted by Beck *et al.* (1993), the part of the procedure of developing a model without any reference to in situ field data is a function solely of 'the knowledge and imagination of the analyst'. Although all researchers follow the same scientific method, the wide range of computer models developed for the same purpose, shows that researchers differ in their knowledge and imagination. The properties of a model developed by a researcher or a group of researchers, depends on the background knowledge of the researchers involved, their niche in the scientific world, and their skills to communicate with software engineers.

Technology. A model representation should be tractable from the technological point of view (Casti, 1998). Whether a model is tractable, depends on the model representation, the tool used to program it, and the hardware. A complex model, concerning many variables defined in multiple dimensions, using a large number of model equations can become intractable, because of its long run time. Although computer power still doubles every year, while more and more optimisation algorithms become available in the

modelling tools, it can be expected that computer power will continue to pose a constraint on model building. Technology is also important regarding the tools used in combination with the dynamic model. These tools determine how the inputs and outputs of a forward model can be analysed, and how field data can be used to optimise the model. Standard visualisation tools included in Geographical Information Systems (Burrough and McDonnell, 1998) can be used to analyse the data in a spatial context, while statistical tools can be used to test hypotheses, to perform interpolations or simulations as input to a dynamic model, or to calibrate model parameters.

1.3.2 The model development cycle

From this discussion of the factors involved in modelling, it can be concluded that, at least in the environmental sciences, a model should not be regarded as a fixed entity, with generic application. Instead, it is a tool which needs to be fashioned to all factors involved in a modelling study, and these factors are specific for the study problem, the technology, and the people involved in a research. So, the activity of model building is crucial for environmental modelling. Finding the optimal model is in most cases a trial and error procedure, as represented by the model development cycle (see also chapter 2, and Jørgensen, 1988) consisting of a sequence of procedural steps (Figure 1.6):

1) model structure identification, involving the selection of the processes governing the behaviour of the system to be modelled, and the mathematical representation of these processes,
2) programming the model, involving the conversion of the mathematical representation of the processes to a computer program,
3) estimation of appropriate values of input variables and parameters using field data, which can be done by upscaling and/or inverse modelling,
4) validation.

In the case of unsatisfactory results in one of these steps, the procedure has to be re-done, starting at a previous step, until the optimal model has been found.

This thesis deals with problems involved in three components of the model development cycle shown in Figure 1.6: 1) computer programming, i.e. creating a computer model according to the concepts defined by the model structure, 2) estimation of inputs and parameters by upscaling, 3) estimation of inputs and parameters by inverse modelling. In addition, a fourth problem is dealt with, which is 4) how to train people to build models according to the model development cycle. Each of these 4 points are discussed in the following sections.

1.3.3 Issues involved in programming the model

Programming involves the conversion of the mathematical representation of the processes in a dynamic model to a computer program. Equation 3, representing the structure of a model, is a generic equation, encapsulating a wide range of different environmental

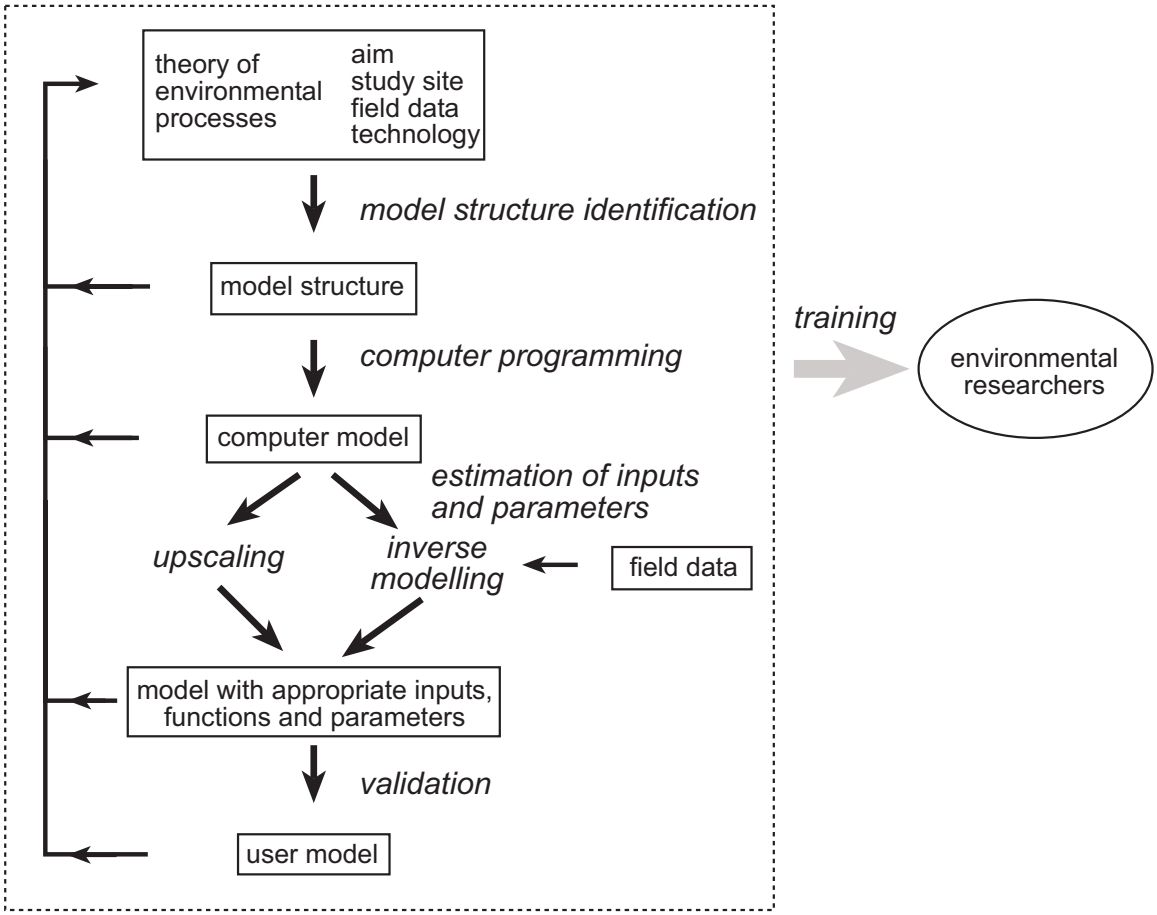


Figure 1.6. Model development cycle.

models. The type of tool which allows programming of all these different models is a *system programming language*, such as Fortran or C++. Another approach is to use a programming language developed for the specific purpose of environmental model building, which is a so called *environmental modelling language*. The main concept of an environmental modelling language is that models are constructed using pre-programmed building blocks that can be combined in a useful way to construct an environmental model. People designing such a language have to choose how much and what kind of functionality needs to be included in the building blocks, resulting in building blocks that allow many models to be constructed by a competent model builder. This means that a balance needs to be found between the advantage of including a lot of functionality in the building blocks and the disadvantage that the language becomes less generic; as more functionality becomes built in, the modeller has less opportunity to modify the basic units. It is clear that the approach defining these building blocks depends on the kind of environmental models that need to be constructed with the language, and for this reason, a number of languages that could be called environmental modelling languages exist. Since most Geographical Information Systems (GIS, c.f., Burrough and McDonnell, 1998) deal with static data, the modelling languages in these systems are designed for the construction of models in the two or three dimensional spatial domain only (e.g. ESRI,

2002; EarthVision, 2002), although some standard GIS include functionality for analysing time series of maps (e.g., IDRISI 2002, GRASS 2002). Other tools focus on dynamic modelling of non spatial data, such as ModelMaker (2002) and STELLA (2002), or are mathematical modelling languages lacking standard functions and visualisation tools for spatial data (e.g., Matlab 2002).

The environmental modelling software package PCRaster (PCRaster 2002; Van Deursen 1995; Wesseling et al. 1996) is an environmental modelling language for building dynamic spatial environmental models, which is here called a *dynamic spatial environmental modelling language*. Since it is both evaluated in this thesis, while new concepts are developed for extensions, it is important to summarize the main concepts here. Starting from equation 3, the following simplifications, which one could equally well call design choices, are made in PCRaster.

Since the set of functions f_k, $k = 1..m$ is too complex to be solved analytically, a regular discretisation is made of the two dimensional domain only, the third spatial dimension is not considered, and the spatial index becomes an index $\mathbf{s}_{rc}$, referring to row and column numbers of the grid cells with an area |u|. Further, it is assumed that functions are used representing the change over a time step Δt, which is the same for all t. The discretised version of the model can be written as

$$z_k(\mathbf{s}_{rc}, t+\Delta t) = f_k\left(z_k(\mathbf{s}_{rc}, t), k = 1..m; i_j(\mathbf{s}_{rc}, t), j = 1..n; \mathbf{s}, t\right) \quad \text{for each } t \qquad (4)$$

In addition, it is assumed in the PCRaster software that f_k can be represented by a combination of standard spatial functions provided by the language, which are the building blocks I referred to at the start of this section:

$$z_k(\mathbf{s}_{rc}, t+\Delta t) = g_1, g_2, .g_n\left(z_k(\mathbf{s}_{rc}, t), k = 1..m; i_j(\mathbf{s}_{rc}, t), j = 1..n; \mathbf{s}, t\right) \text{for each } t \quad (5)$$

These standard spatial functions can be simple, such as the addition of two variables without spatial interaction, or more complicated, such as performing numerical solutions of differential equations, including spatial interactions. In this approach of using standard spatial functions, model building becomes the activity of combining these functions with their proper inputs. In PCRaster, this is done in a sequential so-called dynamic modelling script, which is the program of the model. Such a program has a dynamic section, which is iterated through time. For each time step, a sequence of the standard spatial functions is executed, as shown in Figure 1.7. The environmental model builder needs to define this sequence. For a more detailed description of the concepts, the reader is referred to the next two chapters.

Although environmental modelling languages, and more specifically PCRaster, are useful tools for programming dynamic spatial models, the use of these languages also involves several problems. Two key problems are dealt with in this thesis. First, the approach to provide the model builder with a restricted set of pre-programmed building blocks, comes with several possible disadvantages related to the activity of programming the model, the range of models that can be built with the language, and the performance of the models regarding run times. As a result, their value might be limited for programming a model. From an evaluation of the PCRaster language done in this thesis,

it follows that extensions are needed to provide the modeller with an environmental modelling language that is also capable of dealing with three dimensional models and stochastic models. The difficulties related to the design of such languages is the second problem related to programming a model that is dealt with in this thesis.

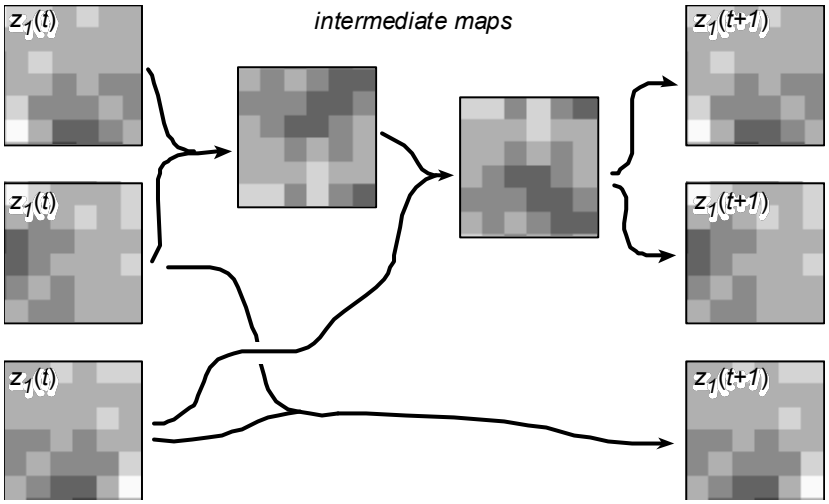


Figure 1.7. Dynamic model in PCRaster. A combination of standard functions (arrows) on raster maps is made describing the change in the model variables between *t* and *t*+1.

1.3.4 Issues involved in upscaling

The issue of upscaling is related to the step of estimating model inputs and parameters in the model development cycle. This step is important, since the model outcome is strongly dependent on the value of the inputs and parameters. In many cases the resolution of the field data and the resolution applied in the model are different, and an upscaling or downscaling procedure needs to be performed in order to use the field data in the model. In upscaling, the inputs and parameters are derived from field data without using the model itself (Figure 1.8).

A short description of the main concepts related to upscaling is given here, for details, the reader is referred to Bierkens *et al.* (2000), Blöschl and Sivapalan (1995), Blöschl (1996), from which all concepts described here are taken. Upscaling and downscaling theory is built around a key concept called support. As noted in section 1.3.3, the spatial domain of a dynamic spatial model, is subdivided (i.e., discretised) into a finite number of sub-areas, with an area |u|, while the temporal domain is subdivided in sub-intervals, with a length Δt . The area of these sub-areas and the length of these sub-intervals is called the *support* of a model. It is the largest area (or volume) and time interval for which the properties represented by a model are considered homogeneous. These sub-areas or sub-intervals themselves are called *support units*. The values of the model inputs, variables

and parameters are representative for the support used in the model, while the functions need to be representative for the change in model variables occurring over a time step, at the support of the model. The term support can also be used for field measurements, representing the area (or volume) and time interval for which the measured properties are considered homogeneous, and for which only the average value is measured and not the variation within. The term *scale* refers to the same concept as support, where a large scale refers to a large support. The term *scale transfer* means changing the support, while *upscaling* and *downscaling* refer to increasing and decreasing the support, respectively. An *upscaling or downscaling method* refers to the procedure describing how to calculate changes in input values, parameters, or a function in a model when the support is changed.

Upscaling and downscaling methods are important for environmental modelling, since the values and the spatial pattern of most environmental attributes, when measured in the field, depend on the support of measurement. As a result, inputs and parameters in environmental models, need to depend on the scale of the model, while in some cases, the model structure (i.e., the functions in the model) needs to change with scale too, since a description of processes appropriate at one scale, does not need to be appropriate at another scale. Many examples illustrating the problem of scale in a wide range of environmental studies and upscaling and downscaling methods to solve this problem are given in Bierkens *et al.* (2000), Blöschl and Sivapalan (1995), Burrough and McDonnell (1998).

In this thesis, the issue of scale is dealt with in two case studies concerning the process of infiltration. To illustrate the problem of scale in infiltration modelling, a small, highly simplified, steady state, example is given. At a small support, typically 0.04 m^2, a reasonable model to describe actual infiltration (A, mm/h) is:

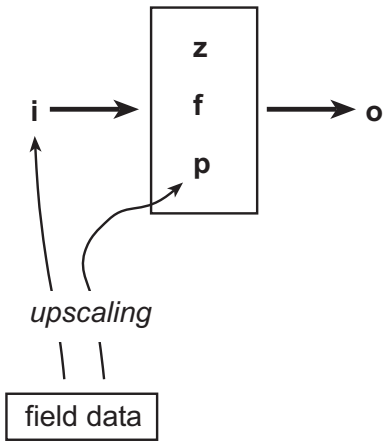


Figure 1.8. The need for scaling when estimating model inputs (**i**) and parameters (**p**) from field data. **z**, model variables; **f**, model functions; **o**, model outputs.

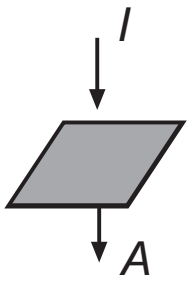


Figure 1.9. Input (*I*, mm/h) and actual infiltration (*A*, mm/h) in an infiltration model.

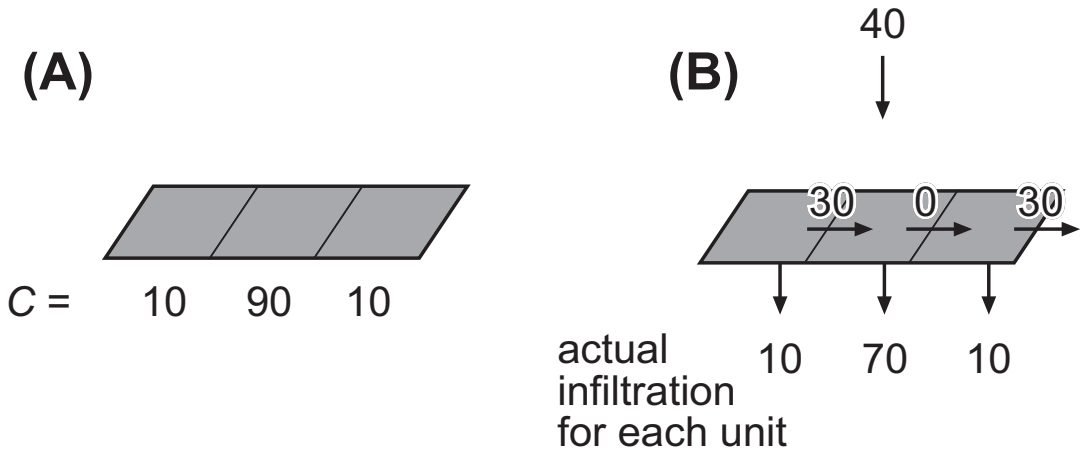


Figure 1.10. (A) Measured infiltration capacity (mm/h) for four neighbouring areas, (B) resulting actual infiltration and inflow fluxes from upstream neighbours with 40 mm/h rain. All units are mm/h.

$$A \begin{cases} = I & \text{for } C > I \\ = C & \text{for } C \le I \end{cases} \qquad (6)$$

with, *I*, the input of water to the soil surface (mm/h) – think of this as rainfall – and, *C*, the infiltration capacity of the soil (mm/h), which is a parameter, see also Figure 1.9. Now, let us assume we have measured the infiltration capacity at one specific location, resulting in C = 30 mm/h. With an input I = 40 mm/h, the actual infiltration can be calculated as 30 mm/h, using equation 6. Now, let's assume the infiltration capacity is known for neighbouring areas of 0.04 m^2, on a small transect, see Figure 1.10A. Is the model given by equation 6 also valid to calculate A for the larger support of this transect? If so, how? The first approach would be to take the average value of the three measurements, assuming that this value can be used at the larger support. With the input I = 40 mm/h, we have A = C = (10+90+10)/3 = 36.33 mm/h. But this is not correct, as shown in Figure 1.10B, because at this larger support, the process of flow of water between the support units needs to be taken into account: equation 6 needs to be applied

for each support unit separately, where *I* becomes the sum of rainfall (mm/h) and inflow from the areas upstream. If we do this, we find an average actual infiltration of 30 mm/h (Figure 1.10B), and we conclude that *C* needs to be 30 mm/h when we want to apply equation 6 at the larger support. It is said that this value for *C* is the *effective* (or *representative*) value, valid at this larger support. But the problem is not solved yet. Doing the same calculation for the larger support using the same infiltration capacity values for the three areas and including the inflow from upstream, but a different value for the rainfall, results in a different effective value for *C*. So, strictly speaking, *C* cannot regarded as a constant parameter anymore. Instead, it needs to be regarded as a variable when *I* becomes variable in time, which is always the case with a rainstorm. This means that under transient conditions, a process description different from equation 6 is needed at the larger scale, which includes this relation between *C* and *I*. This shows that, in addition to change of parameter values with change of scale, the process description may also need to change with scale.

Scale dependency is a general problem with environmental modelling, particularly for the case of dynamic rainfall-runoff modelling. A dynamic rainfall-runoff model simulates the processes involved in rainfall interception by the trees, surface storage of water, infiltration, and drainage of water to an outflow point of a catchment. In the part of the thesis dealing with upscaling, focus is on the development of upscaling methods for upscaling of infiltration measured at a scale corresponding to the small support of 0.04 m^2 (like in the example above) to the support of the rainfall-runoff model used. The number and applicability of upscaling methods currently available is limited, and there is a need for new and better upscaling methods for infiltration (Beven, 1989; Binley et al., 1989; Blöschl *et al*, 1995; Blöschl and Sivapalan, 1995; Harms and Chansyk, 2000). Two

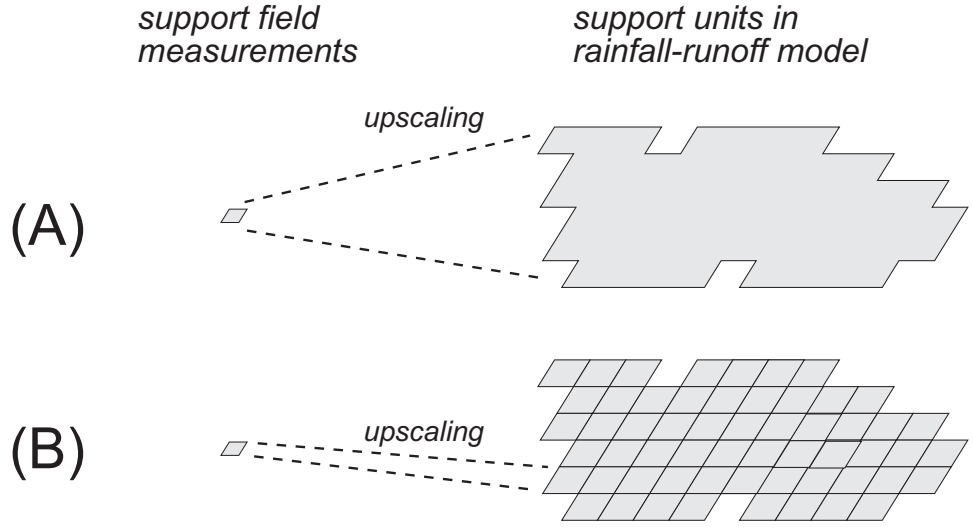


Figure 1.11. Upscaling from the support of field measurements (left) to the support of a rainfall runoff model (right). (A) upscaling to the support corresponding to the area of the catchment in the rainfall-runoff model, one support unit, (B) Upscaling to the support of grid cells used in the rainfall-runoff model, multiple support units.

possible approaches to develop an upscaling method for a dynamic rainfall-runoff model will be followed in this thesis (Figure 1.11). In the first approach, a rainfall-runoff model is used with a support regarding infiltration corresponding to the size of the whole catchment, and the model includes only one support unit, which is the catchment. This means that the parameter(s) regarding infiltration are assumed to be homogeneous within the catchment, although other processes in the model can still be spatially variable. In this approach, an upscaling procedure needs to be developed that scales from the small support of the field measurements, to the support corresponding to the size of the catchment. In the second approach, a rainfall-runoff model is used with a support regarding infiltration corresponding to the size of a grid cell used in the model, and the number of support units corresponds to the number of grid cells in the model. Here, an upscaling procedure needs to be developed that ranges from the support of the field measurements to the support of the grid cells used in the model, for each support unit. Both approaches are expected to suffer from problems caused by the fact that the process of rainfall-runoff is highly transient (Blöschl and Sivapalan, 1995).

1.3.5 Issues involved in inverse modelling for spatial interpolation

When field data on inputs and parameters or appropriate upscaling procedures to estimate inputs and parameters of a dynamic model are not available, inverse modelling is the only method that can be used to estimate inputs and parameters of a dynamic model. In upscaling, model inputs and parameters of a dynamic model are estimated with an upscaling method which is run independently of the dynamic model, using field data on inputs and parameters. Unlike upscaling, inverse modelling estimates the inputs and parameters using field measurements of output variables of the dynamic model and the dynamic model itself, as shown in Figure 1.12. In inverse modelling, it is assumed that the best set of values for the inputs and parameters of a dynamic model corresponds to the set of values resulting in the smallest possible difference between the output of the dynamic model and field measurements of the same output variable(s) (McLaughlin and Townley, 1996). The difference between the output and field measurements is reflected by an *objective function* (sometimes called *goal function*) which is a mathematical procedure to calculate the aggregated difference between a vector of model outputs and field data, where the lowest outcome of the objective function mostly represents the smallest difference between outputs of the dynamic model and field data.

A procedure of inverse modelling comprises an iteration of three steps: 1) select a set of inputs and parameters for the dynamic model, 2) run the dynamic model with this set of inputs and parameters, 3) calculate the value of the objective function. The iteration is stopped when the set of inputs and parameters is found with the lowest value of the objective function. The number of iterations needed can be reduced when results of previous iterations are used in a better selection (i.e., expected to result in a low value of the objective function) of inputs and parameters in step 1. Different procedures to do so are described in Beasley *et al.* (1993), Falkenauer (1998). When inverse modelling is restricted to finding parameter values only instead of both inputs and parameters, it is also known as *calibration*. When the aim of inverse modelling is mainly to make model

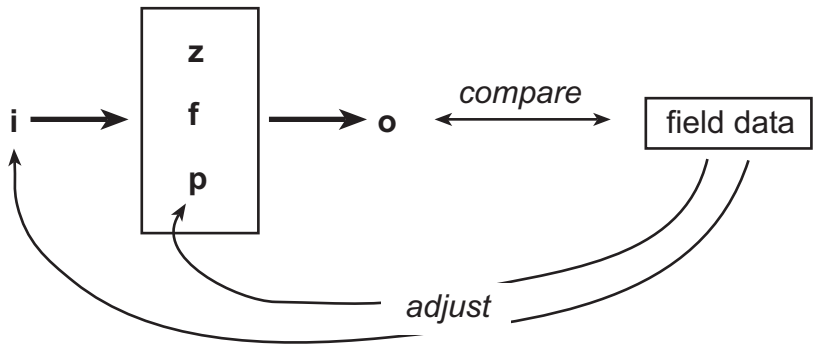


Figure 1.12. Estimating model inputs (**i**) and parameters (**p**) from field data by inverse modelling. **z**, model variables; **f**, model functions; **o**, model outputs.

outputs *exactly* fit field data, the term *conditioning* or *data-assimilation* is widely used for inverse modelling.

One of the issues in inverse modelling is to minimise the computer time required in an inverse modelling procedure. Computer run times can be large, since the dynamic model needs to be run for each iteration in the procedure of inverse modelling. In general, the computer time needed for an inverse modelling procedure can be expected to be dependent on a wide range of issues: the amount of field data used for inverse modelling, the objective function and the minimisation of its outcome required, the capability of the dynamic model to simulate environmental processes at the study site in a proper way, the size of the input and parameter space in which the optimal input and parameter values need to be found, the run time of the dynamic model, and the inverse modelling procedure applied. These issues are related to most of the factors in environmental model building (Figure 1.4), and inverse modelling will only be successful when all these factors are dealt with, in an integrated approach.

The issue of computer run time needed in an inverse modelling procedure is important in a case study (Chapter 7) dealt with in this thesis. This case study tries to predict the sedimentary architecture in three dimensions using an inverse modelling procedure with a dynamic model simulating the erosional and depositional processes in a river system occurring over time spans of thousands of years. The main output of the dynamic model is a prediction of the sediment type for each location in three dimensions. The field data used are observations of the sediments in a number of wells (boreholes) and the aim is to find the input values for the dynamic model resulting in a sediment type predicted by the model that corresponds to the field data at the observational locations. Running the model with these input values results in a prediction of the sedimentary architecture in three dimensions, where the predicted sediment type at observational locations corresponds to the measured type. Nowadays, this prediction (or interpolation) of sediment type in between wells is accomplished mainly using methods that imitate the structure of the deposit, without direct use of the knowledge of the processes that formed the deposit, resulting in predictions that are not always realistic. When the issue of computer run time related to inverse modelling with a dynamic model can be overcome, it is expected that predictions can be made which are more realistic.

1.3.6 Issues involved in training model building

While model building itself comes with many difficulties, as has been pointed out in the previous sections, teaching model building to people who are more or less unfamiliar with environmental models might even be more difficult. This is mainly because training involves many different subjects and people to be trained, under different learning environments. The subjects of training comprise all procedural steps of the model development cycle (Figure 1.6), in all phases of learning, while the people to be trained are heterogeneous regarding their background knowledge, disposition, culture, and method of education they are used to. The learning environment can be a class room situation with lectures by experts, computer practicals with back up from tutors, or distance learning, with support of students provided at a distance, through the world wide web. A good training programme needs to reckon with all these situations of training.

This is only possible with adequate tutors, and appropriate learning materials and tools, which need to be maintained and updated by their authors. The learning materials need to include standard textbooks, software manuals, and computer exercises. In addition, software tools are needed that can be used by students in each of the procedural steps of the model development cycle (Figure 1.6). This is an important issue, since students in the environmental sciences are mostly not experienced in programming. As a result, they may encounter problems when using software that requires users with a lot of background in informatics. So, tools are needed matching the conceptual thought processes of environmental scientists, which allows students to focus on learning model building, instead of learning informatics. For instance, in the procedural step of programming the model in the model development cycle (Figure 1.6), the use of environmental modelling languages is likely to be efficient, since these provide easy to use building blocks for construction of models. For distance learning over the internet, additional tools are needed providing alternatives for direct evaluation and communication between the student and the tutor in a class room situation.

The environmental modelling language PCRaster (see Section 1.3.3) comes with course material and tools (PCRaster, 2002) for teaching dynamic spatial environmental model building, both in a classroom situation, and through distance learning. In this thesis, the question will be answered how efficient this material is for training model building, and which improvements are needed.

1.4 Research questions

1.4.1 Central research question

In order to guarantee that dynamic models built in the future will fulfil all requirements of good science, all steps in the model development cycle (Figure 1.6) need to be supported by appropriate methods and theories from science and technology, while training tools are needed to teach people how to perform all steps in the model development cycle. From this viewpoint, a central question is: does existing technology and/or science provide sufficient means for (training) all steps in the model development cycle for construction of dynamic spatial environmental models? To answer this question, all steps

in the model development cycle would need to be evaluated, for all factors in involved in dynamic modelling (Figure 1.4), and for all types of dynamic spatial models, which is not feasible in the framework of a thesis. Instead, a restriction will be made to dynamic spatial models simulating spatially continuous fields only, while a selection of issues related to the model development cycle is studied, evaluating only the issues of 1) programming, 2) upscaling, 3) inverse modelling, and 4) training, as shown in Figure 1.13. The central research question is:

Is the current state of computer technology and science sufficient for executing the steps of programming, upscaling, and inverse modelling in the model development cycle, and for teaching all steps in the model development cycle, with respect to dynamic spatial environmental models simulating continuous fields?

The first three issues of programming, upscaling and inverse modelling are procedural steps in the model development cycle. They are all important, since a weakness in one of these steps will affect the model development cycle as a whole, resulting in a model with

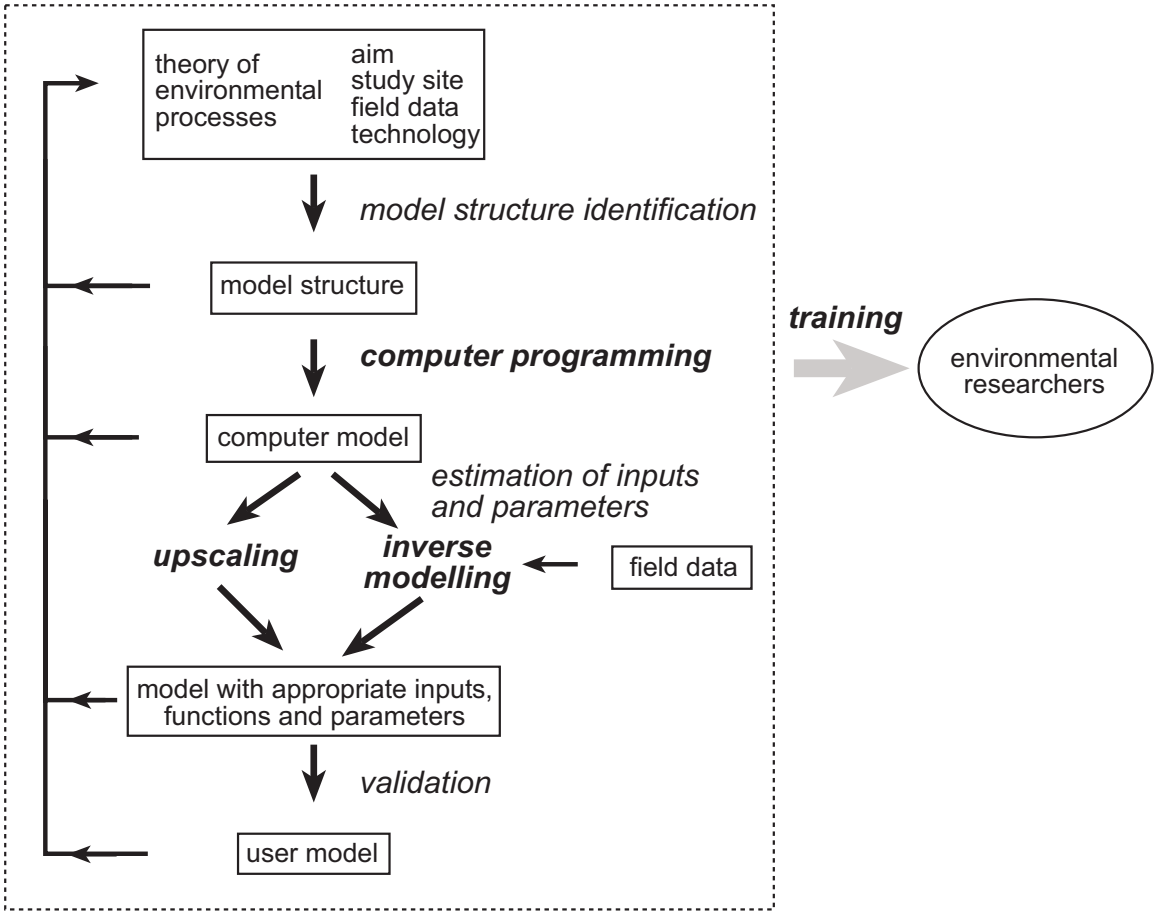


Figure 1.13. Model development cycle. Issues of programming, scaling, inverse modelling and teaching (in bold type) represent those dealt with in this thesis.

a quality below the quality it could have had. The fourth issue of training the model development cycle involves teaching the steps of programming, upscaling, and inverse modelling, but also teaching the other steps in the model development cycle, which are the steps of model structure identification and validation (Figure 1.13).

For all issues studied in the thesis, there is a need for technology and science, apart from other needs which are not considered here, such as the need for human intelligence, or the need for field data. Although the issues of programming, upscaling, inverse modelling and training all need technology and science, the role of technology and science is different for each issue. Although a full separation between technology and science is not always possible, since they overlap, the research questions are grouped in those mainly related to science, and those mainly related to technology.

The research questions related to the model development cycle are given in a somewhat arbitrary order, since there is no hierarchy in these steps: programming, upscaling, inverse modelling. The questions related to the issue of training model building are given thereafter, since they can only be answered after answering the research questions regarding the model development cycle itself. In the research questions defined below, the term model refers to a dynamic spatial environmental model simulating spatially continuous fields.

1.4.2 Programming: research questions

The step of programming involves the conversion of the mathematical representation of the processes in a model to a computer program of the model (Section 1.3.3). Since it involves the use of computer software, it is strongly related to existing software technology, and the first research questions evaluate how good this technology is for the purpose of programming the model. From the limitations that follow from the evaluation of existing technology, research questions are posed regarding possible solutions offered by scientific concepts for development of new software. The research questions are:

Questions related to software technology:

Are the concepts included in dynamic spatial environmental modelling languages better than those of system programming languages, for programming the model?

What are the restrictions of existing dynamic spatial environmental modelling languages for executing the procedural step of programming in the model development cycle?

Questions related to science:

Can we extend dynamic spatial environmental modelling languages with (concepts for) functions for efficient programming of three dimensional models?

Can we extend dynamic spatial environmental modelling languages with (concepts for) functions for efficient programming of stochastic models in order to calculate error propagation in dynamic spatial models?

1.4.3 Upscaling: research questions

Field data are used to estimate inputs and parameters in a model. Upscaling involves scaling methods needed to change the support of field data to appropriate values of inputs and parameters at the support used of the model (Section 1.3.4). The issue of upscaling is not dealt with in a general way, treating all methods of upscaling, for all possible situations. Instead, upscaling is evaluated using two case studies involving upscaling methods for infiltration. Apart from research questions related to the specific issue of upscaling infiltration, some research questions are posed related to the general issue of upscaling. The answers to these questions will be inferred from the knowledge and experience gained from the study into infiltration. As a result, the answer to these more general research questions related to upscaling will be somewhat restricted.

The issue of scientific methods for upscaling is still a major research question in science, and the research questions to science are posed first. Since it may be possible, that scientific methods for upscaling are, or will become available as standard software tools, technological issues are treated in the second group of research questions.

Questions related to science:

As noted in section 1.3.4, scaling of infiltration is dealt with by upscaling to 1) the support of a catchment (Figure 1.11A) and, 2) the support of the units in a rainfall-runoff model (Figure 1.11B). The research questions related to these issues are given in this order, followed by one research question regarding upscaling in general:

Upscaling to the support of a catchment:

Is it possible to define an upscaling method to scale infiltration measured at a small support (appr. 0.04 m^2) to effective values of infiltration for catchments (1-7500 m^2), which correspond to values derived from measurements at that larger scale, under steady state conditions of rainfall, runoff and infiltration?

Does an upscaling method, which scales infiltration measured at a small support (appr. 0.04 m^2) to effective values of infiltration for catchments (1-7500 m^2), give better results when applied to a dynamic rainfall-runoff model than using the same model without the transfer function?

Upscaling to the support of the units in a rainfall-runoff model:

When an upscaling method scaling infiltration measured at a small support (appr. 0.04 m^2) is used to derive effective values of infiltration for model units (appr. 100 m^2) in a dynamic rainfall-runoff model, does this rainfall-runoff model give better results regarding discharge from a hillslope (appr. 7500 m^2) and a catchment (appr. 0.4 km^2) than these found when the upscaling method is not used?

Upscaling in general:

Is the existing software technology sufficient for solving problems of upscaling related to estimating inputs and parameters of a rainfall-runoff model, and in other upscaling situations?

Questions related to technology:

Is the existing software technology sufficient for solving problems of upscaling related to estimating inputs and parameters in a rainfall-runoff model, and in other upscaling situations?

1.4.4 Inverse modelling: research questions

Inverse modelling is a means to estimate inputs and parameters of a model by comparison of a set of outputs of a model with measurements of these outputs. As was noted in section 1.3.5, one of the important issues in inverse modelling is to minimise the computer time required in an inverse modelling procedure. This issue involves both science, for instance the methodology of inverse modelling procedures, and technology, for instance computer run time needed to run a model in an iteration of the inverse modelling procedure. Just like upscaling, inverse modelling is dealt with in a case study, to which most research questions relate. The results of this case study are put in a general context, by answering research questions regarding inverse modelling in general.

Questions related to science:

Does existing scientific knowledge provide sufficient means to make predictions of three dimensional sedimentary architecture with a dynamic spatial model, conditioned to observations?

Does existing scientific knowledge provide sufficient means to do inverse modelling with dynamic spatial environmental models?

Questions related to technology:

Does existing computer technology provide sufficient means to make predictions of three dimensional sedimentary architecture with a dynamic spatial model, conditioned to real-world observations?

Does existing computer technology provide sufficient means to do inverse modelling with dynamic spatial environmental models, in practice?

1.4.5 Training: research questions

Training model building involves teaching all steps of the model development cycle (Figure 1.13) to people with some background in environmental sciences, but without, or with little, background in model building. As noted in section 1.3.6, teaching can only be done with appropriate course materials, modelling software, and, in the case of distance learning, software for running courses over the internet. These software tools and course materials are included in the PCRaster environmental modelling software. It is this set which is evaluated in the thesis, aiming at answering the following research question:

Does the existing PCRaster environmental modelling language, its associated course material and tools for distance learning, provide an efficient means for teaching dynamic spatial model building in all phases of education, for a wide range of people?

1.5 Thesis outline

The thesis consists of four parts, where each part focuses on one of the four issues explained above. The first part covers the step of programming in the model development cycle. Chapter 2 evaluates existing dynamic spatial environmental modelling languages, by a comparison with other programming languages. I try to resolve two of the restrictions of existing dynamic spatial modelling languages in Chapter 3 and 4. These chapters describe a new prototype dynamic spatial modelling language with extra functionality, in addition to the functionality of existing dynamic spatial modelling languages. Chapter 3 describes the concepts used for three dimensional modelling in this language, while Chapter 4 describes how error propagation using a stochastic modelling language is done.

The second part of the thesis deals with the issue of upscaling, by describing two case studies into upscaling of infiltration. Chapter 5 describes upscaling of infiltration from the local scale to the catchment scale, while Chapter 6 deals with upscaling from the local scale to the scale of individual model units of a dynamic spatial model. The third part of the thesis covers inverse modelling, dealt with in the case study of Chapter 7, which is a dynamic spatial model simulating sedimentary disposition and erosion.

The fourth part of the thesis focuses on training model building, with a review chapter on the PCRaster software and course materials, and their use in training students (Chapter 8). Chapter 9 gives the conclusions.

1.6 References

Abbott, M.B., J.C. Bathurst, J.A. Cunge, P.E. O'Connel & J. Rasmussen (1986), An introduction to the European Hydrological System - Systeme Hydrologique "SHE", 1: History and philosophy of a physically-based distributed modelling system. Journal of Hydrology 87, pp. 45-59.

Beasley, D., D.R. Bull & R.R. Martin (1993), An Overview of Genetic Algorithms: Part 1, Fundamentals. University Computing 15, pp. 58-69.

Beck, M.B., A.J. Jakeman & M.J. McAleer (1993), Construction and evaluation of models of environmental systems. In: Beck, M.B., Jakeman, A.J., McAleer, M.J. (eds.), Modelling change in environmental systems, John Wiley & Sons Ltd., New York, USA.

Beven, K.J. (1989), Changing ideas in hydrology - the case of physically-based models, Journal of Hydrology 105, pp. 157-172.

Beven, K.J. (2000), Uniqueness of place and the representation of hydrological processes, Hydrology and Earth System Sciences 4, pp. 203-213.

Binley A. & K. Beven (1989), A physically based model of heterogeneous hillslopes: 2. Effective Hydraulic Conductivities, Water Resources Research 25, pp. 1227-1233.

Bierkens M.F.P, P.A. Finke & P. de Willigen (2000), Upscaling and downscaling methods for environmental research. Dordrecht: Kluwer.

Blöschl, G. & M. Sivapalan (1995), Scale issues in hydrological modelling: a review. Hydrological Processes 9, pp. 251-290.

Blöschl, G. (1996), Scale and scaling in hydrology. Wien: Technische Universität Wien, Institut für Hydraulik, Gewässerkunde und Wässerwirtschaft. (Wiener Mitteilungen, band 132)

Blöschl, G., R.B. Grayson & M. Sivapalan (1995), On the representative elementary area (REA) concept and its utility for distributed rainfall-runoff modelling. Hydrological Processes 9, pp. 313-330.

Blöschl, G., & M. Sivapalan (1995), Scale issues in hydrological modelling: a review. Hydrological Processes 9, pp. 251-290.

Burrough, P.A. & R.A. McDonnell (1998), Principles of Geographical Information Systems. Oxford: Oxford University Press.

Casti, J.L. (1998), Would-Be Worlds: How Simulation Is Changing the Frontiers of Science. New York: Wiley.

De Wit, M.J.M. & E.J. Pebesma (2001), Nutrient fluxes at the river basin scale. Part II: the balance between data availability and model complexity. Hydrological Processes 15, pp. 761-775.

Donnelly-Makowecki, L.M. & R.D. Moore (1999), Hierarchical testing of three rainfall-runoff models in small forested catchments, Journal of Hydrology 219, pp. 136-152.

EarthVision (2002), info at: http://www.dgi.com

ESRI (2002), Environmental Systems Research Institute, info at: http://www.esri.com/

Falkenauer, J. (1998), Genetic algorithms and grouping problems. New York: Wiley.

GRASS (2002), info at: http://www.geog.uni-hannover.de/grass/

Gurney, W.S.C. & R.M. Nisbet (1998), Ecological Dynamics. New York: Oxford University Press.

Harms, T.E. & D.S. Chanasyk (2000), Plot and small-watershed scale runoff from two reclaimed surface-mined watersheds in Alberta. Hydrological Processes 14, pp. 1327-1339.

Hasbargen, L.E. & C. Paola (2000), Landscape instability in an experimental drainage basin. Geology 28, pp. 1067-1070.

Heuvelink, G.B.M. (1998), Error Propagation in Environmental Modelling with GIS. London: Taylor & Francis.

IDRISI (2002), info at: http://www.clarklabs.org/

Jørgensen, S.E. (1988), Fundamentals of ecological modelling. Amsterdam: Elsevier.

Jothityangkoon, C., M. Sivapalan & D.L. Farmer (2001), Process controls of water balance variability in a large semi-arid catchment: downward approach to hydrological model development. Journal of Hydrology 254, pp. 174-198.

Klemes, V. (1983), Conceptualisation and scale in hydrology, Journal of Hydrology 65, pp 1-23.

MATLAB (2002), info at: http://www.mathworks.com/

McLaughlin, D. & L.R. Townley (1996), A reassessment of the groundwater inverse problem. Water Resources Research 32, pp. 1131-1161.

ModelMaker (2002), info at: http://www.modelkinetix.com/modelmaker/
PCRaster (2002), info at: http://www.geog.uu.nl/pcraster
Reggiani, P., M. Sivapalan & S.M. Hassanizade (1998), A unifying framework for watershed thermodynamics: balance equations for mass, momentum, energy and entropy, and the second law of thermodynamics. Advances in Water Resources 22, pp. 367-398.
Sprague, R.H. & H.J. Watson (eds.) (1982), Decision support systems: putting theory into practice. London: Prentice-Hall.
STELLA (2000), info at: http://www.hps-inc.com/edu/stella/stella.html
Toffoli, T. (1989), Cellular automata machines, Cambridge, Massachusetts: MIT Press.
Van der Perk, M. (1997), Effect of model structure on the accuracy and uncertainty of results from water quality models. Hydrological Processes 11, pp. 227-239.
Van Deursen, W.P.A. (1995), Geographical Information Systems and Dynamic Models. Utrecht: Koninklijk Nederlands Aardrijkskundig Genootschap/Faculteit Ruimtelijke Wetenschappen, Universiteit Utrecht.
Wesseling, C.G., D. Karssenberg, W.P.A. van Deursen & P.A. Burrough (1996), Integrating dynamic environmental models in GIS: the development of a Dynamic Modelling language. Transactions in GIS 1, pp. 40-48.

PART I:

COMPUTER PROGRAMMING

2 THE VALUE OF ENVIRONMENTAL MODELLING LANGUAGES FOR BUILDING DISTRIBUTED HYDROLOGICAL MODELS

Derek Karssenberg

Abstract: An evaluation is made of the suitability of programming languages for hydrological modellers to create distributed, process-based hydrological models. Both system programming languages and high level environmental modelling languages are evaluated based on a list of requirements for the optimal programming language for such models. This is illustrated with a case study, implemented using the PCRaster environmental modelling language to create a distributed, process-based hydrological model based on the concepts of KINEROS-EUROSEM. The main conclusion is that system programming languages are not ideal for hydrologists who are not computer programmers because the level of thinking of these languages is too strongly related to specialised computer science. A higher level environmental modelling language is better in the sense that it operates at the conceptual level of the hydrologist. This is because it contains operators that identify hydrological processes that operate on hydrological entities like 2D maps, 3D blocks and time series. The case study illustrates the advantages of using an environmental modelling language as compared with system programming languages in fulfilling requirements on the level of thinking applied in the language, the reuse-ability of program code, the lack of technical details in the program, a short model development time, and learnability. The study shows that environmental modelling languages are equally good as system programming languages in minimising programming errors but are worse in generic application and performance. It is expected that environmental modelling languages will be used in future mainly for development of new models which can be tailored to modelling aims and available field data.

Published as: Karssenberg, D., The value of environmental modelling languages for building distruted hydrological models. Hydrological Processes, in press. Reproduced with permission.

2.1 Introduction

Since the 1980s several major hydrological research groups have been developing distributed process-based hydrological models for simulating the transport of water, soil, nutrients and pollutants. Examples are groundwater transport models (e.g. AQUA3D, 2001; Harbaugh and McDonald, 1996; Zheng, 1990), rainfall-runoff models (e.g. SHE, Abbott, 1986a,b; TOPMODEL, Beven, 1997; LISFLOOD, De Roo *et al*., 2000), rainfall-runoff models including erosion (e.g. Grayson *et al.*, 1992; EUROSEM, Morgan *et al.*, 1998; Tucker *et al*., 1999), and rainfall-runoff models with nutrient or pollutant transport (e.g. Mackay and Ban, 1997). A single internet search on hydrological + modelling

delivers hundreds of responses. Clearly, hydrologists have a continuing need for new and better models, since concepts on how to represent hydrological processes in computer simulation models are still evolving. This change of ideas in modelling is being driven by new observation techniques, including remote sensing, and data storage and presentation technology such as Geographical Information Systems (GIS), that provide larger volumes of useful data than ever before. As with other areas of science such as astronomy or biology, new methods of data collection and processing may improve scientific understanding in ways that were not possible before they were introduced.

The development of new numerical models has always been restricted by the functionality of programming languages and computer power. This will continue to be so in the future, since model demands on computers and programming languages increase, due to a further refinement of model concepts, larger data sets, and a wider application of calculation intensive methods such as Monte Carlo simulation. Although computers have opened up new research fields in hydrology, at the same time they pose restrictions, and the history of hydrological modelling has been influenced by the capacity of computers and the languages to program them.

In the 20th century, we saw a gradual transition from stand alone programs for hydrological modelling, developed by small research groups who were at the same time developers and users, towards off-the-shelf computer programs with a user friendly interface, linked to GIS, for a generic wide application. Initially, there was little unity in hydrological modelling and every research group wrote their own software, in system programming languages such as C++ or FORTRAN.

As time went by, the number of models that was worked on reduced as hydrologists selected a small set of models that embodied good science and straightforward implementation. These are the models that have been linked to GIS. As noted above, GIS can be used to supply much information for hydrological modelling, ranging from digital elevation models of the land surface to time series of groundwater levels and river flows. Hydrologists started to link these GIS databases to their models so that the input and output aspects of hydrological modelling could be simplified and visualised, and the results placed in a spatial and temporal context. Standard models were coupled to GIS following the loose coupling or tight coupling approach (Burrough, 1996). Loose coupling involved ad hoc, manual exchange of data between a model with a proprietary GIS. Since manual data exchange is susceptible to errors, software for automatic data exchange was written and some GIS firms began to hard-wire the hydrological models into their systems, which was called tight coupling. This provided hydrologists with a ready-to-use modelling tool which was much favoured by consultants, but not by scientists. The reason for the disfavour by scientists is that the process-understanding and algorithms are rarely state-of-the-art and also that the program code is usually inaccessible or difficult to change in such systems.

Since these standard hard-wired models are of limited value for scientific research, hydrologists wishing to develop new models are left with two options. The first approach is to develop new models from scratch, or to use blocks of code from others, via a system programming language, and to link it to an existing GIS. The second, more recent approach, is to use an environmental modelling language (EML) running inside a GIS, which is known as embedded coupling (Burrough, 1996). Unlike system programming languages, which are generic purpose languages, EML are higher level programming

languages with a specific application domain, in our case hydrological model construction. The approaches adopted by EML are along the following lines (Wesseling et al, 1996a):

1) provide a set of operators operating on spatio-temporal data in which widely accepted generic hydrological processes have been coded using accepted, clearly understood algorithms,
2) provide these operators in a suitable way that they can be glued together in a model by a hydrologist using his or her hydrological understanding, rather than computer expertise,
3) embed this set of tools for model construction in a GIS-like software environment providing data base management and generic visualisation routines for the spatio-temporal data read and written by the model,
4) provide standard interfaces to other programming languages so that new or alternative operators can be added by the user in ways that are fully compatible with the EML.

The range of responses to the challenge of developing EML has been large, although most of them do not fulfil all four concepts given above. Some hydrologists (e.g. Olsthoorn, 1998) have used spreadsheet programs for modelling, a step that Campbell (1985) termed a "revolution in groundwater modelling", or technical computing languages such as MATLAB (MATLAB, 2001). Although very powerful, such languages lack an embedded coupling with a GIS. Others developed graphical modelling languages with an easy to use interface for model construction (ModelMaker, 2001; STELLA, 2001). These are very powerful for process modelling, but their non-spatial operators do not provide sufficient functionality for hydrological modelling. Modelling languages included in many GIS have the advantage that they come with powerful database and visualisation tools and that they are per definition spatial. On the other hand, the modelling languages of proprietary GIS (e.g. ESRI, 2001) are too much focused on database management and static operations, to fulfil the requirements for spatio-temporal hydrological modelling. The most interesting developments, however, have been those made by specialist groups who have created languages along the four concepts of EML given above. These include products such as GRASS (GRASS, 2001), PCRaster (PCRaster, 2001; Van Deursen, 1995) and Simile (Simile, 2001). The number of hydrologists using these EML is small compared to those using system programming, partly because of their more recent development. But EML have proven their usefulness in model building, and it is time for an objective evaluation of this addition to the hydrologists' tool box.

Therefore, this chapter will evaluate the concepts and application of EML for hydrological model building. The outline is as follows: first, a set of requirements of a programming language for hydrological model construction is defined. Second, a case study illustrates the creation of a runoff model similar to the hydrological component of KINEROS-EUROSEM (Morgan et al., 1998), and demonstrates how EML differ from system programming languages. The construction of the case study model in EML is described using PCRaster. By evaluating this software implementation of the runoff model, EML are compared with system programming languages on the basis of the

requirements of a programming language for hydrologic model construction given in the first part of the chapter.

2.2 A programming language for hydrological model building: requirements

There is a continuous need for new and better models, since concepts on how to represent hydrological processes in computer simulation models are still evolving. Models evolve partly because of changing ideas in hydrology, but also because of the expanding availability of input data, and the increasing capability to handle them using GIS, as noted in the introduction. Since field data are important inputs to models, but also crucial for model calibration and validation, it is necessary that the process descriptions in the model are tailored to the available data. Models ignoring important processes which can be fed with input data will be too simple, while models representing processes for which no data are available will be unnecessarily complex (Van der Perk, 1997). In addition, the model process descriptions need to be tailored to the aims of modelling. Modelling the peak discharges of a river system could be done with a lumped, or simple spatial model in many cases, while predicting surface runoff and erosion needs a more complex spatial model. Since both the data available and the aim of the model will be different for each study, it could be said that each study needs a new model, or at least a model which can be modified compared to models developed for previous studies. This does not mean that each model is unique in all its details, since there are concepts in hydrology which have proven their generic application, like interception equations or surface water routing techniques, which are applicable in many models. The main challenge in model building is to find the optimal generic process representations and an appropriate way to combine these for a specific purpose.

If we look at model building in more detail, model development can be regarded as a process whereby different candidate model structures are evaluated until an optimal model formulation is found (Van Deursen *et al*., 2000). Most modelling studies involve such a model development cycle (Figure 2.1), although it is mostly not described in reports. But there are examples of studies describing or explicitly focusing on such a comparison between different model structures (e.g., Van der Perk, 1997; Donelly-Makowecki and Moore, 1999; Grayson et al., 1992). The model development cycle involves three phases per candidate model (Figure 2.1). In the first phase, the mathematical description of the model is defined, based on knowledge of hydrological processes and how they interact in the study area. It contains the mathematical equations simulating the set of hydrological processes. The computer program of the model, written in the second phase, is the numerical representation of this mathematical description. In the third phase, this computer model is tested by evaluating whether it fulfils the aim of the study or not. This involves mostly calibration of the model with field data. This phase may reveal weaknesses in the model. If weaknesses are found, attempts are made to improve the next candidate model structure: the mathematical description of the model is redefined, and a new program is written and tested against field data. Evaluating better and better candidate models is continued until a model is found which fulfils the aims of the modelling study.

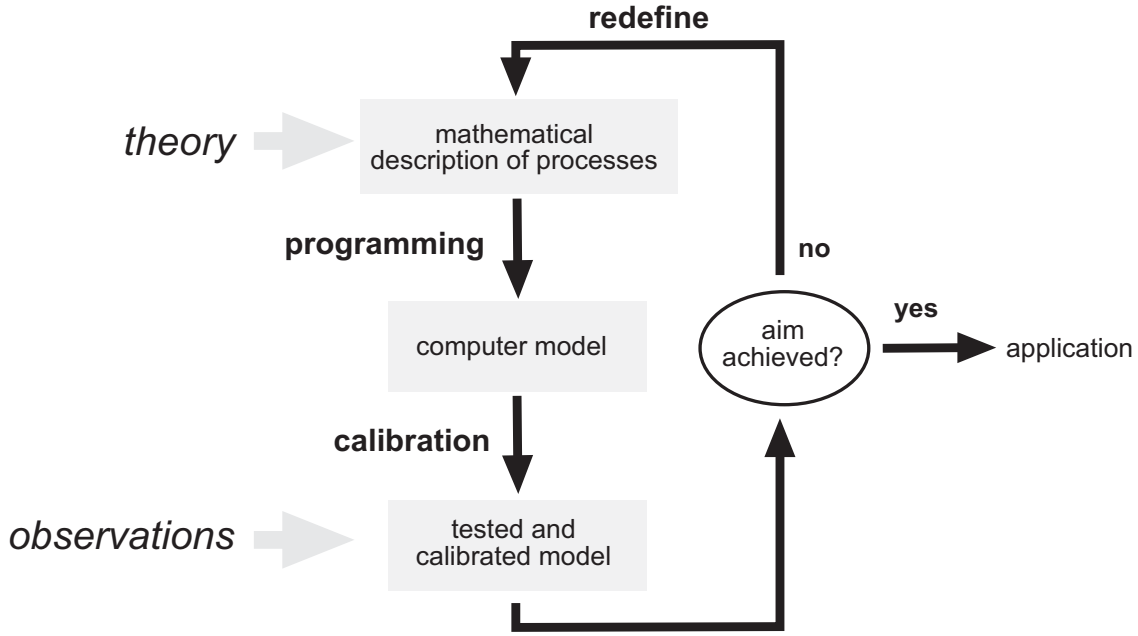


Figure 2.1. Model development cycle.

The second phase in the model development cycle involves programming. It is here that the choice of the programming language becomes important. The choice of the programming language is currently not a key issue in scientific literature, probably because model implementation is regarded as a technical detail, unrelated to hydrology. But if we look at Figure 2.1, we see that the language used for model implementation may have major impact on the results of a modelling study since it affects the whole model development cycle. For instance, if it is possible with the language to change the model without too much programming, it allows evaluation of a larger number of different, candidate models, simply because it is practically more feasible. Also, the efficiency of converting the mathematical description of processes to a program code of the computer model is highly dependent on the computer language used. If the computer language is difficult to handle by hydrologists, specialist programmers are needed for software implementation, and evaluating different candidate models becomes the work of a team of programmers and hydrologists, where the hydrologist has to explain to the programmer how each candidate model should be programmed. Instead, a modelling language that can be used by the hydrologist would permit prompt software implementation of a new mathematical description of a process by the hydrologist, allowing for interactive changing the model and evaluating its output. Other issues related to the choice of the language are also important here, such as performance of the model, or the chance of errors in the code.

For judging between different programming languages, a list of requirements for the optimal computer programming language for hydrological model development is needed. Based on a list of criteria for computer languages in Highman (1967), and with the model development cycle in mind, the following list of requirements for a language for hydrologic model construction has been formulated.

1) *Level of thinking of hydrologists*. The level of thinking of a hydrologist should be represented in the model program code of models. If the concepts of the computer language represent those of hydrology, it allows easy conversion of the mathematical description of processes to program code. Also, if the program code resembles hydrological concepts, the language is more accessible to hydrologists. As a result, hydrologists can program models themselves, without the need for specialist programmers. In addition, a language operating at the level of thinking of hydrologists enables easy exchange of program code between researchers since programs can be read by hydrologists.

2) *Reuse of program code*. Different hydrological models use similar standard operations and algorithms for the simulation of processes. For each process there are many programs that perform the same computations according to identical equations and algorithms described in the literature. For making a new model that needs a different combination of processes it should be possible to re-use and combine the program code of existing models. A programming language is needed that allows reuse of blocks of already written code simulating specific processes (e.g., GMS, 1998; Harbaugh and McDonald, 1996; Leavesley *et al*., 1998). For model development, it should be easy to link these blocks together without the burden of complex programming.

3) *Generic approach to common problems*. It should be possible to make any type of distributed process-based model with the language, including (sub-) models for simulating environmental processes related to process-based hydrological modelling such as plant growth, received solar radiation and land degradation.

4) *No technical computer details*. Most developers of hydrological models are hydrologists, not computer programmers. So a programming language for model development should relieve the researcher from technical computer details that would distract her or him from scientific research.

5) *Short development time*. A programming language resulting in shorter development times would allow modifications to existing models or construction of completely new models within the framework of one modelling study, thereby tailoring the model to the modelling aims and available data. From the viewpoint of the model development cycle (Figure 2.1), a short development time is even more important. If it were possible to construct a new candidate model by changing an existing candidate model without too much programming work, it would allow evaluation of more different candidate models.

6) *Minimising programming errors*. The possibility of making programming errors in hydrological model development should be as low as possible. Errors easily occur since hydrological models are large and complex; having many different processes being simulated by complicated numerical solution schemes. It should be easy to detect these errors.

7) *High performance*. Since distributed hydrological models use large data sets and computationally intensive algorithms, execution times can be a problem and a

programming language is needed that minimizes computation time. This requirement does not have the highest priority since the performance of computers is still doubling every 2 years.

8) *Easy to learn for hydrologists*. The programming language should be easy to learn for hydrologists who do not necessarily have programming experience.

2.3 Mathematical definition of the runoff model

The case study used in this chapter for the evaluation of different programming languages is a runoff model aiming at simulating the effect of different patterns and directions of runoff pathways in agricultural catchments on the shape of the hydrograph at different locations. The model should describe the processes of interception, surface storage, infiltration and runoff with process-based equations, since detailed, spatially and temporally distributed, field data are available, including maps of preferential runoff pathways over fields. Most of these processes are included in KINEROS-EUROSEM (Morgan *et al*., 1986) but this model does not simulate preferential runoff directions on fields caused by agricultural operations. So a new model has been constructed that includes data on the pattern of runoff pathways over fields. All other process descriptions are taken from EUROSEM, schematised in Figure 2.2.

For each time step, the net rainfall (P_n, m per time step) reaching the ground is:

$$P_n = P - c \cdot C \tag{1}$$

with; P, the open field rainfall in meters per time step; c, percentage cover of the vegetation (m^2/m^2); and C, the amount of water transported to the interception store, in meters per time step for the area covered with vegetation. C is (*Merriam*, 1973):

$$C = S_m \left(e^{\left(\frac{-P_{c,t-1}}{S_m} \right)} - e^{\left(\frac{-P_{c,t}}{S_m} \right)} \right), \tag{2}$$

with; S_m, maximum content of interception store (m); $P_{c,\,t-1}$, cumulative precipitation since start of the rainfall, at preceding time step (m); $P_{c,t}$, cumulative precipitation (m) since start of the rainfall, at current time step. Potential infiltration is simulated with the Smith and Parlange equation (Smith and Parlange, 1978) additionally accounting for rock fragments in the soil (Woolisher *et al.*, 1990):

$$F = K \cdot \frac{e^{F_c/B}}{e^{F_c/B} - 1}, \tag{3}$$

with; K, effective saturated hydraulic conductivity of the field (m per time step); F_c, cumulative infiltration since start of the rain (m); B, saturation deficit parameter modified for rock fragments (m); F, potential amount of infiltration in a time step (m per time

step). Water is routed through rill areas with the kinematic wave using the Manning equation (Li *et al*., 1975; Chow *et al*., 1988).

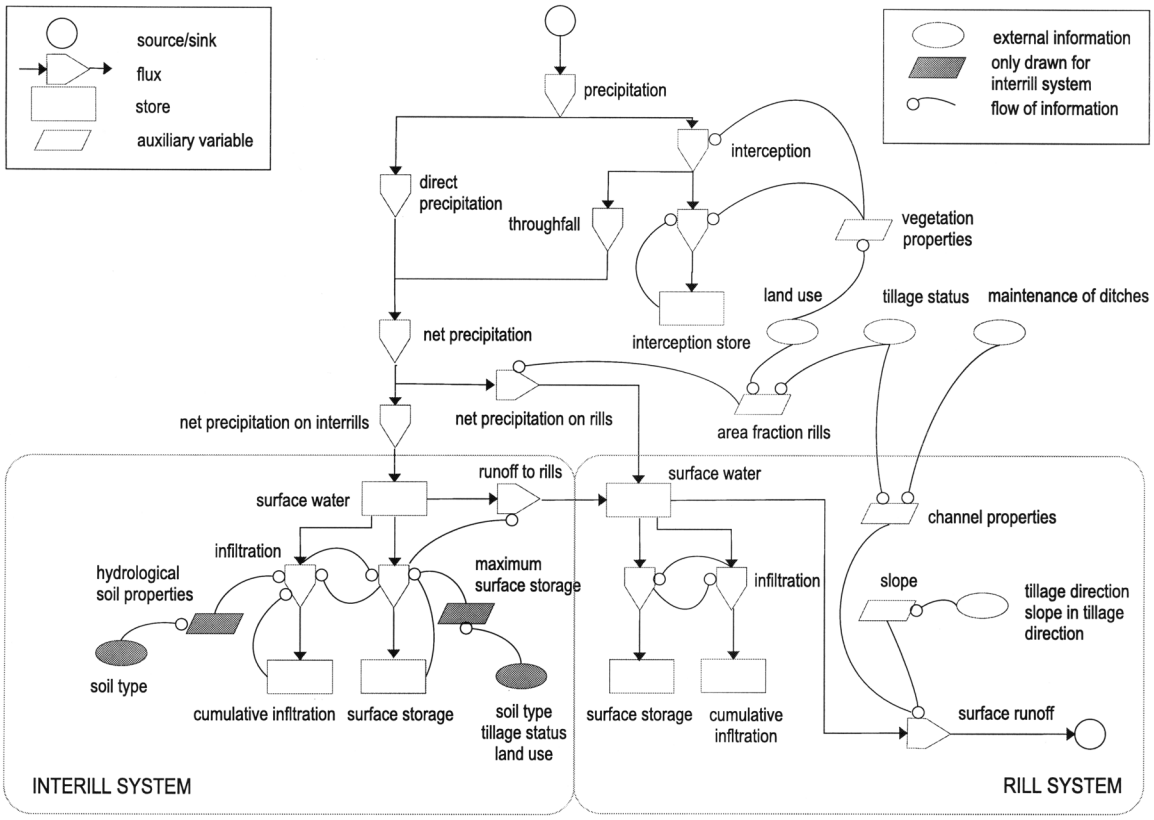


Figure 2.2. Flow diagram of the runoff model.

2.4 Environmental modelling language for hydrological modelling

What is the best programming language to construct an event based runoff model for the case study described above? For answering this question, languages can be compared by looking at the entities that are changed by the principal statements (operators) of the language and the type of functionality that is provided by the operators. Both the kind of entities and the functionality of the operators should 1) be compliant to the kind of objects that will be changed by the language and what changes need to be made to them, and 2) represent the thinking level of the user of the language. Table 2.1 gives the entities and the functionality of some widely used programming languages. It shows that some languages represent most aspects of the computer in their entities and operators, so called

Table 2.1. Entities and functionality of operators of some programming languages. Top: lower level languages, bottom: higher level languages.

Language	Entity	Functionality of operators
Assembly languages	bits	changing bits
System programming languages (e.g. Fortran, C++)	integers, floating points, arays	adding, summing, looping
General purpose scripting languages (e.g. Tcl/Tk, Python)	strings, integers, arrays	adding, summing, looping
Standard Query Language	tabular data	selecting data from a table, ordering
Technical computing languages (e.g. Splus, Matlab)	matrices, floating points	matrix inversion, adding matrices, calculating statistics
Graphical modelling systems (e.g. ModelMaker, Stella)	non spatials states, fluxes	fluxes between states
Environmental modelling languages (e.g., PCRaster, Idrisi)	maps, timeseries, blocks	summing maps, iterating through time, topological links, transport of water, visualisation

low-level languages, while others deal with entities and operators that are specific for a certain application field, so called high-level languages. Assembly languages, being low-level languages, are not efficient for environmental model construction since virtually every aspect of the computer has to be defined in the program. System programming languages and generic scripting languages are at a higher level, since each operator represents several machine instructions that would need a block of program code if written in an assembly language.

The entities of system programming languages such as strings and floating points are not at the level of thinking of a hydrologist, neither do they represent the objects of study in hydrology. In a program for a hydrological model, entities are needed that represent hydrological objects such as landscapes (maps), below surface composition (3D blocks), and time (time series), while operations at these entities are needed that represent hydrological processes. To illustrate this, assume that the interception equation (1) has to be implemented with a system programming language. Much program code would be needed for defining data structures and file formats of spatio-temporal data, iterations for defining the time, a spatial 'multiply' operator operating for each map unit. It would be more convenient to use a programming language that sets the loop and the operations in just two statements:

```
timer 100
 save I = P * c
```

where `timer` defines iterations saying that the statement below the timer has to be executed for 100 time steps. The variables `I`, `P`, `c` are spatial (maps) or non-spatial entities defined implicitly in the language. The operator '`*`' multiplies two spatial entities and the 'save' operator means the result should be saved for each time step. So the statements mean multiply map `P` with map `c` resulting in the map `I`, and do this for each time step. Another example is the kinematic wave transport of water needed for the model. Kinematic wave transport with the Manning equation is a more complicated operation than multiplication but it is also generic and involves only a few input map entities (see Li *et al*., 1975; Chow *et al*., 1988):

$$Q_t = \mathrm{f}(Dir, Q_{t-1}, QIn, \alpha, \beta, T, D),$$

with:

f	kinematic wave operator,
Dir	map with directions of flow,
Q_{t-1}	map with water discharge in direction *Dir* in previous time step (e.g., m^3/s),
QIn	map with addition or subtraction of water to/from the flow (e.g., m^3/s),
α	map with coefficient (Li *et al*., 1975; Chow *et al*., 1988),
β	map with coefficient (momentum or Boussinesq coefficient, Chow *et al*., 1988),
T	time step (e.g., s),
D	map with distance of flow to downstream unit in direction *Dir*,
Q_t	map with water discharge in direction *Dir* at current time step (e.g., m^3/s).

In a system programming language, this operation takes several pages of code to be implemented since it is a spatial operation that needs a numerical solution of the kinematic wave equations. The higher level definition in an EML is one line of code as one operation operating on spatial entities (maps) as shown above. This use of higher level entities and operators is the main concept of EML.

The practical application of this depends upon the generic nature of the entities and operators in the EML. A relatively limited number of entities and operators should support a wide range of models to be constructed, for many different hydrological situations. Simple operations like loops and mathematical operations on maps make up the main part of most models and are generic. The same seems to hold for more complex operations like kinematic wave transport or groundwater flow. Stable numerical solution techniques for most of these operations have been described in papers and standard textbooks (e.g. Bear and Verruijt, 1987; Chow, 1988; Zheng, 1993; Olivera and Maidment, 1999) and can be regarded as methods that have proven their general application.

High level statements for hydrological modelling can only be made with EML that have built in knowledge of both spatial and temporal entities, supported by hydrological operators such as kinematic wave transport. Examples of such EML are GRASS (GRASS, 2001), PCRaster (Van Deursen, 1995; Wesseling *et al*., 1996a; PCRaster, 2001), Simile (Simile, 2001) and concepts described by Takeyama (1997).

2.5 Implementation of the runoff model with PCRaster

PCRaster has been used to illustrate how the runoff model can be implemented in an EML. Entities in PCRaster are series of raster maps for spatio-temporal attributes, time series for temporal non-spatial data and lookup tables. Map entities are assigned a type according to their content in hydrological/geographical context. Types used are Boolean, nominal and ordinal for classified data, scalar and directional for continuous data, and local drain direction representing drainage networks (Figure 2.3). The PCRaster language contains 125 operators operating on these entities. Operators included are non-spatial (point) operations, spatial operations (Burrough and McDonnell, 1998) and spatio-temporal time operations for reading and writing temporal data, e.g. hydrographs at specific locations. The concept of the language is similar to mathematical thinking and notation. As in mathematics, each operator in PCRaster solely affects the resulting variable of that operator and has no side effects on other variables in the program, which might be the case in system programming languages. Additionally, the syntax obeys mathematical notation. For example, creating a map containing for each cell the total

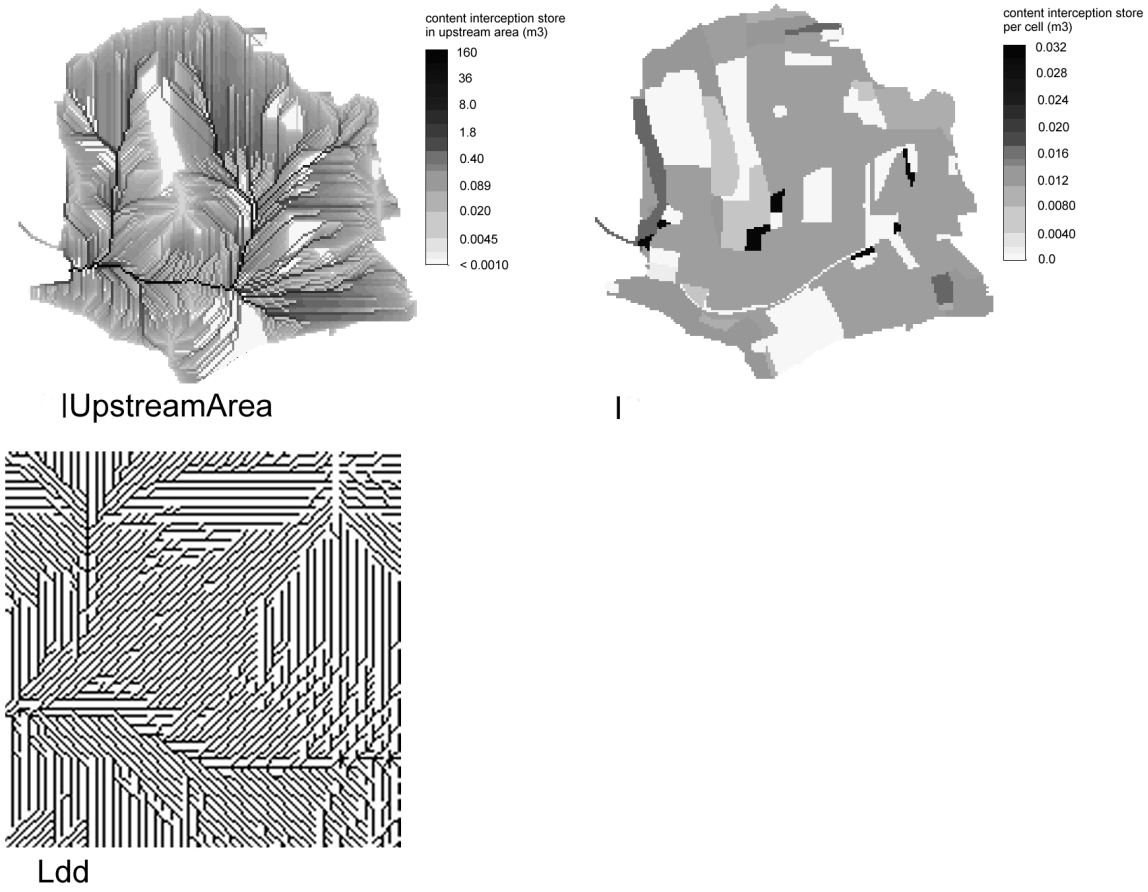


Figure 2.3. Input and output maps of `IUpstreamArea = catchmenttotal(I,Ldd);` `IUpstreamArea`, infiltration in upstream area; `I`, infiltration per cell; `Ldd`, local drain direction map (zoomed area), lines represent flow directions**.**

amount of a variable in its upstream area is done with the 'catchmenttotal' operator of PCRaster. For infiltration:

```
report  IUpstreamArea = catchmenttotal(I,Ldd);
```

where the 'catchmenttotal' operator has two inputs, the map `I` with the amount of infiltration for each cell and the map `Ldd`, the local drain direction map defining the drainage pattern. The operator generates a new map (here named `IUpstreamArea`) containing for each cell the total amount of infiltration in its catchment (Figure 2.3). In the implementation of the runoff model budget checks on catchment scale are performed by applying this operation on all water fluxes (e.g. precipitation, infiltration, surface storage) and summing results.

A PCRaster program (script) is structured in different sections (Table 2.2). The binding is a list of program variables linked to filenames. The initial section initialises the model by setting initial states and constant values of variables using the PCRaster operators. In the case study model, operations in this section calculate constant parameters like channel dimensions and roughness, interception parameters, infiltration parameters and initial states like soil water content. The dynamic section is an iterative sequential section and loops for the number of time steps defined in the timer section. It contains operations for the temporal behaviour of the model: interception, infiltration, surface storage, kinematic wave transport. The result of each separate operation can be saved with the report keyword for each time step or a selection of time steps.

GIS-like database and visualisation software for spatio-temporal data using the file formats of the PCRaster language is integrated with PCRaster in one system. The case study model uses this software for scenario studies in cultivated land in the Ouvèze river basin (S. France). The database consists of rain time series files, 10 maps and 35 lookup tables that are directly read by the EML program. PCRaster supports prompt visualisation of its entities with map and time series display programs. Model results were significantly different for different patterns of runoff pathways, as Figure 2.4 shows (cf. Van Dijck, 2000). An extensive description of the model is available at WWW http://www.geog.uu.nl/pcraster/runoff/.

2.6 Evaluation of environmental modelling languages

Based on the example of implementing the runoff model with PCRaster, it is possible to make a judgement about the usefulness of such an EML for hydrological model construction. This assessment is done per requirement given in the introduction and a comparison is made with system programming. Table 2.3 provides a judgement based on the discussion below.

BINDING

```
binding
 # time of one timestep (s)
 T=scalar(5);
 # map with locations of discharge measurements
 Flumes=report.map;
 ...
```

Links program variable names to input and output maps, timeseries, and tables

INITIAL

```
initial

 ...
```

interception parameters

```
 # nr of channels per metre, wheel tracks; ploughing
 report ChanPMWh=lookupscalar(CahanPMWhTBL,Landuse);
 report ChanPMPl=lookupscalar(ChanPMPlTBL,Landuse);
 ...
```

parameters for runoff routing

```
 ...
```

infiltration parameters and initial soil moisture. interrill & rill areas

```
 ...
```

surface storage parameters. interrill & rill areas

TIMER

```
timer
 # number of timesteps
 1 8500 1;
```

number of timesteps (iterations) for dynamic section

```
 # report at every timestep
 all=1+1..endtime;
 # report at first and each 50th timestep
 anim=1,50+50..endtime;
 ...
```

defining timesteps variables area saved in report functions in dynamic section

DYNAMIC

```
dynamic

 # rain (m/T)
 Pr=timeinputscalar(PTSS,Clone);
```

precipitation, reading rain time series

```
 # intercepted water (m/T)
 report(anim) Int=Pr*COV;
 # amount in interception store (m)
 ICSt=ICSStM*(1-exp(-PCum/ICStM));
 ...
```

interception

```
 ...
```

infiltration and surface storage, interrill areas

```
 ...
```

infiltration and surface storage, rill areas

```
 ...
 # water added to the streamflow (m3/s)
 QIn=q*A_R/(T*DCL);
 # discharge (m3/s)
 report(all) Q=kinematic(Ldd,Q,QIn,Alpha,Beta,T,DCL);
 # simulated hydrograph (m3/s) at measurement locs
 report HydrographTss = timeoutput(Q,Flumes);
 # water depth (m), trapezoidal channels, abc formula
 H=(-NBw+sqrt(NBwSq+ZNFour*Alpha*(Q**Beta)))/ZNTwo;
 # wetted perimeter (m), multiple channels per cell
 P=NrCh*(Bw+2*H*SqrtOnePlusSqrZ);
 # alpha
 report(anim) Alpha=AlpTerm*P**AlpPow;
 ...
```

routing surface water in rill areas

```
 # cumulative rill actual infiltration in upstream area (m3)
 FcSub_R=catchmenttotal(FcACCumV_R,Ldd);
 ...
 # total budget (rain minus sum of all fluxes should be 0)
 BudTot=PSub-(ICSub+FcSub_I+FcSub_R+DStSumb_I+DStSub_R+QrCum);
```

budget checks

Table 2.2. Program script of the model, text behind # are remarks. A selection of code is shown, code left out is indicated with '...' . Text in blocks describes left out or partly shown program code.

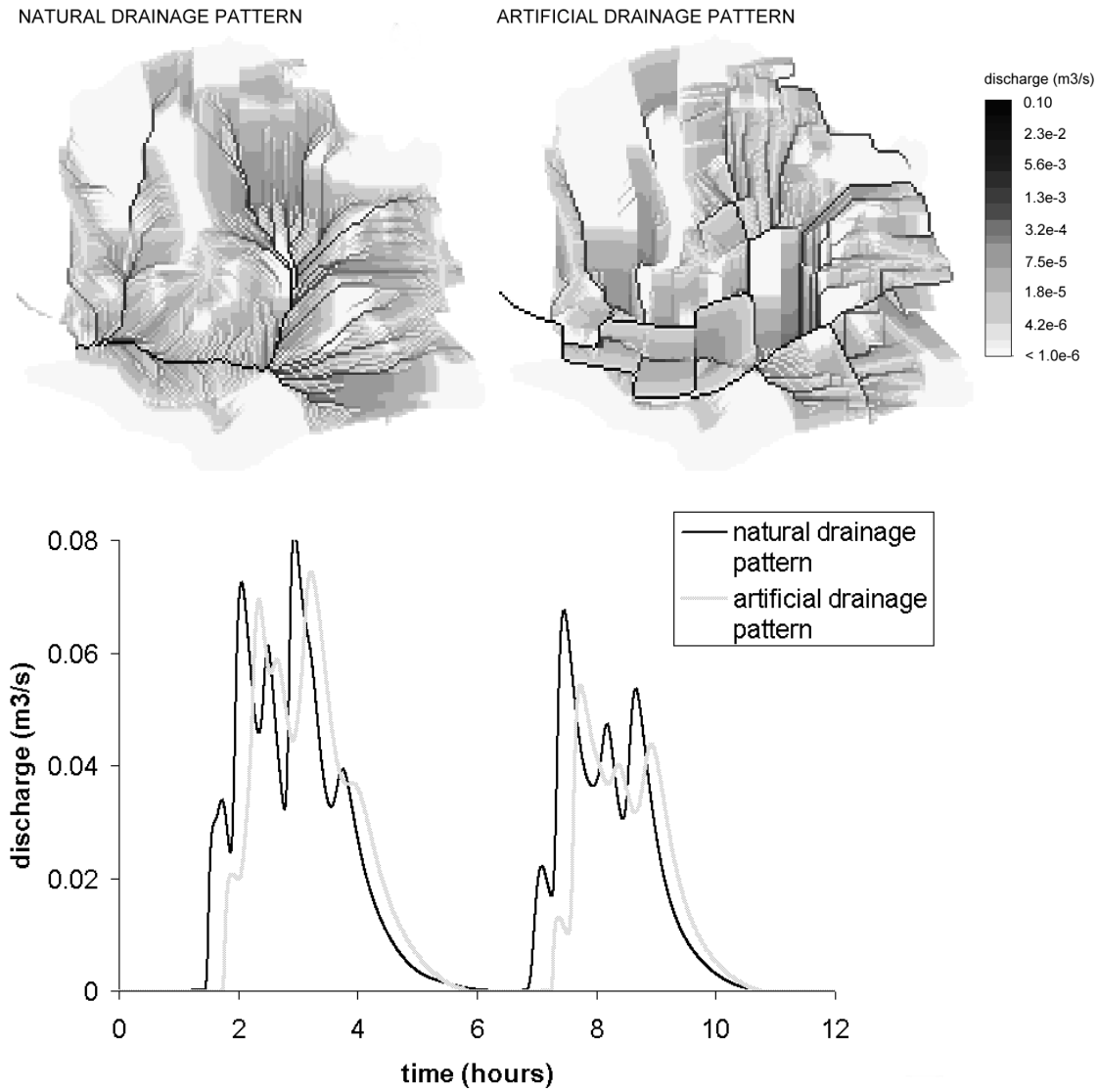


Figure 2.4. Runoff for one rain event, effect of drainage pattern on discharge. Top: runoff after 3 hours; left, natural drainage pattern; right, artificial drainage pattern. Bottom: runoff at outflow point left on maps, natural drainage pattern and artificial drainage pattern.

1) Level of thinking of hydrologists

Unlike system programming languages using entities and operators at the level of a computer, the EML uses entities and operators representing hydrological attributes, dimensions and processes. In addition, hydrologists are familiar with the mathematical notation used. This higher level of thinking of EML opens two possibilities. Unlike system programming languages, the EML can quickly teach environmental researchers without specialist programming knowledge to develop models. Additionally models can be easily worked on by different researchers because they can read each other's programs. This is more difficult for system programs that generally can be read only by the specialist programmer in a research team.

Table 2.3 Fulfilment of requirements for distributed hydrological modelling by system programming languages and PCRaster: +, mainly fulfilled; +/– partly fulfilled; –, not fulfilled.

	System programming languages	PCRaster environmental modelling language
Level of thinking of hydrologists	–	+
Reuse-able	+/–	+
Generic	+	–
No technical computer details	–	+
Short development time	–	+
Minimising programming errors	+/–	+/–
High performance	+	+/–
Easy to learn for hydrologists	–	+

2) Reuse of program code

The EML consists of higher level generic operators simulating hydrological processes which are combined to program the model. The development of these operators has been done in a system programming language by specialist programmers. So, from the technical point of view, combining operators in an EML means gluing together and reusing the program code that is behind these operators. The case study illustrates that this reuse is large. The model in the PCRaster language is only 340 lines of code while the amount of C++ code behind the PCRaster operators used is at least 15,000 lines.

Combining different blocks of code in a system programming language (e.g., Leavesley *et al*., 1998) is in principle also possible, but needs the use of a style guide which should exactly be followed by the programmer. Instead, an EML program implicitly provides standardisation of the components combined, such as standard discretization of time and space in standard file formats. This implicit standardisation allows automatic checks performed by the EML if operators are combined.

3) Generic approach to common problems

As noted in the introduction, an EML captures generic aspects of the hydrological processes in a standard set of operators. The PCRaster language, for instance, provides a means to write models for other fields of hydrology and geomorphology (Eleveld, 1999; Van Deursen and Kwadijk, 1993; De Roo *et al*., 2000), and fields such as radio ecology (Van der Perk *et al*., 2000), plant ecology (Van Deursen and Heil, 1993; Kessel, 1999), ornithology (Van Langevelde, 2000) and veterinary science (Brama *et al.*, 2000). Development of distributed models in PCRaster allows integrated modelling of (sub-) systems of the environment. Integration of different spatial-temporal processes, whether human, hydrological, ecological, geomorphologic, toxicological, is possible in one orderly computer program (e.g. Van der Perk *et al.*, 2000).

A major restriction of EML is that the model builder is restricted by the concepts of the language and the operators provided. The PCRaster language, for instance, can be regarded as generic for simulating hydrological processes in 2D space, although the user has to submit him or herself to the raster based approach in modelling. In addition, the current language has many restrictions that are not shown by our case study. It does not support three-dimensional spatial entities and operations such as groundwater flow

neither does it include standard functionality for stochastic modelling. This is also not supported by other EML. Since the development of EML still continues, it can be expected that such restrictions will be solved in the next 10 years, since concepts for these kinds of extensions exist (e.g., Pebesma *et al.*, 2000; Karssenberg *et al.*, 2000; chapter 3 and 4). But even with these additions, there will be always models that cannot be built with an EML, simply because its set of operators provided is not extensive. To solve these problems, users need to be able to program their own functions operating on maps in a system programming language, which can be made available as a standard operator in the EML. This increases the application range of the language. This plugging in of user made operators is already possible in systems like GRASS and PCRaster.

4) No technical computer details
Since EML are high level languages, technical computer details do not need to be defined in an EML program. For instance, storing maps for each time step is done by preceding a standard statement with the 'report' keyword in PCRaster, something that would take more code and format definitions if a system programming language would be used.

5) Short development time
Evaluating different model structures and finding the optimal model for a specific research project is possible with EML, since alternative models can be built by recombining the standard operators of the language. Immediate post-modelling visualisation of model results is possible with the standard visualisation routines operating on the entities (e.g., maps) used in the language. As a result, models can be reconstructed, run, and visualised in an almost interactive way, where it is easy to evaluate the model results for different alternative model structures. System programming languages do not support this interactive approach since these do not provide the scripting framework of EML to recombine and glue together operations. The short development time is illustrated by our rainfall-runoff model which took one environmental researcher without specialist programming knowledge two weeks to program in the PCRaster language, with testing. Including and testing different interception equations and removing the interrill-rill process description for another modelling study took us 2 days. These development times are short compared to model development in a system programming language. This difference is mainly caused by the high level character of the EML. Operators tested by developers and users of the EML are applied that would take weeks to be developed from scratch in a system programming language.

6) Minimising programming errors
An EML embodies methods for reducing programming errors that are not in system programming languages. By programming in an EML high level operators are combined that have been developed by professional programmers and tested in past research projects. For instance, the numerical solution implemented in the kinematic wave operator of PCRaster has been improved several times throughout the past 5 years, resulting in a stable, reliable operation with generic application. Someone who writes a model using this operator implements 5 years of research experience in the program. The data typing mechanism in PCRaster prevents the user from carrying out nonsense operations since each operator is checked on its input and output. In preventing errors, it

is important that the programmer has a good overview of the complete program and its inputs and outputs, which is guaranteed by the relatively short PCRaster scripts. In addition, intermediate model results in a scripting program can easily be stored on hard disk, by adding a save operator to a line, and these can be visualised with the embedded visualisation tools. So models can be checked step by step by reporting such intermediate results.

This immediate visualisation is the main debugging mechanism provided. Environmental modelling languages do not come with an automatic debugger, which is provided by system programming languages. This is a major disadvantage of EML, and it would be a useful addition, especially if it provides an automatic mechanism to check budgets in hydrological models, too. Currently, the modeller has to define budget checks explicitly in the environmental modelling program. This is quite well possible using the standard operators, but may involve quite an amount of work. The case study model includes 40 lines of code for checking all budgets.

7) High performance
Programs written by professional programmers in system programming languages will usually have shorter execution times than those written in EML. This is mainly because EML are scripting languages that have to perform more run time checks than system programming software (Ousterhout, 1998). In addition, programs written in system programming languages can be optimised, where EML do not have many possibilities for optimisation since fixed operators are glued together. On the other hand, the performance of an environmental modelling application tends to be dominated by the performance of the separate operators, which are typically implemented in a system programming language by professional programmers. This means that in some cases an EML program may be almost equally fast as a system programming language program. Still, EML models will generally be slower than models developed in system programming languages. Optimisation of a system programming language version of a model originally developed in PCRaster decreased run times by a factor 8 (Wesseling *et al.*, 1996b).

8) Easy to learn for hydrologists
An EML can be easily learned because the level of thinking of the language corresponds to the way of thinking of hydrologists (chapter 8, published as Karssenberg *et al*., 2001). The functionality of each of the operators in the language can be taught by using these operators from the command line, without using the iterative program. The next step, modelling in time, is to combine the operators in a program. By following this learning path, scientists can learn to use these languages in one or two weeks. This is a short learning curve compared to system programming languages, since for using these, understanding is needed of computer details.

2.7 Future implications for hydrological modelling

Since EML are easy to learn for people without knowledge of programming, the threshold is low for hydrologists who wish to use them. But many researchers will still prefer system programming languages. The choice of the programming language will

mainly depend on the kind of model that needs to be constructed. It can be expected that development of existing models will be done in the programming language (mostly a system programming language) in which these were originally developed, since rewriting these models in an EML would be inefficient for developers feeling at ease with the language they have always been using, and there is not much reason for rewriting these models in a different language anyway. Also for new models, system programming languages can be more efficient than EML. This holds mainly when performance of the model is important or when models need to be developed with concepts and functions which are not supported by an EML, although most EML come with a facility to plug in new operators, as noted above.

The strongest aspect of EML, and the main reason why they will have impact on future hydrological research, is their ease of use for constructing new models and to quickly modify existing models. In addition to currently existing standard models with fixed process equations, models worked on by many people will be developed that are under continuous change, since an EML provides code that can easily be read and changed by many people who wish to tailor models to their research. Working on a model with several people is also possible with system programming languages, but these languages do not provide an explicit standardization of how to represent hydrological processes in computer code. Such standardization is provided by EML.

Also, EML are a strong tool in the process of finding the optimal model, represented by the model development cycle (Figure 2.1), since modification of the model takes little time, and multiple models can be tried out in a certain research study. This opens the possibility to tailor a hydrological model to the available input data and aims of a modelling study. Such model adjustment is also possible using a system programming language, although it would take much more time.

2.8 Conclusions

Environmental modelling programming languages are conceptually attractive for hydrological model construction because they use entities and operators pitched at the level of thinking needed for writing numerical hydrological models. The case study with PCRaster shows that, compared to system programming, existing EML are better in reuse-ability of program code, lack of technical details in the program, short development time and learnability. They are equally good or worse than system programming languages in minimising programming errors, generic application and performance. In future, it is expected that the application of system programming languages will move to more specialised models, while EML will be used to construct continuously evolving models where tailoring to study aims and field data is important.

Acknowledgements
The runoff modelling study was supported by the European Centre for Geomorphological Hazards. The case study runoff model was developed by the author in cooperation with Simone van Dijck (Utrecht University) to whom thanks are done for cooperation and the dataset. Peter Burrough, Thom Bogaard, Edzer Pebesma, Kor de Jong, Marc Bierkens

(Utrecht University), Willem van Deursen and Cees Wesseling (PCRaster Environmental Software) are acknowledged for comments on the draft manuscript.

2.9 References

Abbott, M.B., J.C. Bathurst, J.A. Cunge, P.E. O'Connel. & J. Rasmussen (1986a), An introduction to the European Hydrological System - Systeme Hydrologique "SHE", 1: History and philosophy of a physically-based distributed modelling system. Journal of Hydrology 87, pp. 45-59.

Abbott, M.B., J.C. Bathurst, J.A. Cunge, P.E. O'Connel & J. Rasmussen (1986b), An introduction to the European Hydrological System - Systeme Hydrologique "SHE", 2: Structure of a physically-based distributed modelling system. Journal of Hydrology 87, pp. 61-77.

AQUA3D (2001), AQUA3D, Groundwater flow and contaminant transport model. Vatnaskil Consulting Engineers. http://www.vatnaskil.is/aqua3d.htm (Access date: 02/10/2001).

Bear, J. & A. Verruijt (1987), Modelling groundwater flow and pollution. Theory and applications of transport in porous media. Dordrecht: Reidel Publishing Company.

Beven, K.J.. (Ed.) 1997. Distributed Modelling in Hydrology: Applications of TOPMODEL. Chicester: Wiley.

Brama. P.A.J., J.M. Tekoppelle, R.A. Bank, D. Karssenberg, A. Barneveld & P.R. van Weerden (2000), Topographical mapping of biochemical properties of articular cartilage in the equine fetlock joint. Equine Veterinary Journal 32, pp. 19-26.

Burrough, P.A. (1996), Opportunities and limitations of GIS-based modeling of solute transport at the regional scale. In D.L. Corwin & K. Loague, eds., Special SSSA Publication Application of GIS to the Modeling of Non-Point Source Pollutants in the Vadose Zone. American Society of Agronomy.

Burrough, P.A. & R.A. McDonnell (1998), Principles of Geographical Information Systems. Oxford: Oxford University Press.

Campbell, A.M. (1985), Discussion on the paper “Groundwater modelling without special programs”, by T.N. Olsthoorn. Ground Water 23, pp. 236-237.

Chow, V.T. & D.R. Maidment, L.W. Mays (1988), Applied Hydrology. New York: McGraw-Hill.

De Roo, A.P.J., C.G. Wesseling & W.P.A. van Deursen (2000), Physically based river basin modelling within a GIS; the LISFLOOD model. Hydrological Processes 14, pp. 1981-1992.

Donnelly-Makowecki, L.M. & R.D. Moore (1999), Hierarchical testing of three rainfall-runoff models in small forested catchments. Journal of Hydrology 219, pp. 136-152.

Eleveld, M. (1999), Exploring coastal morphodynamics of Ameland (the Netherlands) with remote sensing monitoring techniques and dynamic modelling in GIS. Amsterdam: the Netherlands (PhD Thesis Universiteit van Amsterdam, Amsterdam).

ESRI (2001), Environmental Systems Research Institute. Info at: http://www.esri.com/ (Access date: 19/09/2001).

GMS (1998), Groundwater Modelling System v2.1. Reference Manual. Engineering Computer Graphics Laboratory (ECGL), Brigham Young University. Info at: http://www.emrl.byu.edu/ (Access date: 02/10/2001).

Grayson, R.B., I.D. Moore & T.A. McMahon (1992), Physically Based Hydrologic Modelling 1. A Terrain-Based Model for Investigative Purposes. Water Resources Research 28, No. 10: 2639-2658.

GRASS (2001), http://www.geog.uni-hannover.de/grass/ (Access date: 11/09/2001).

Harbaugh, A.W. & M.G. McDonald (1996), User's documentation for MODFLOW-96, an update to the U.S. Geological Survey Modular Finite-Difference Ground-Water Flow Model. U.S. Geological Survey.

Highman, B. (1967), A comparative study of programming languages. Computer monographs 2. New York: Elsevier.

Karssenberg, D., P.A. Burrough, R. Sluiter & K. de Jong (2001). The PCRaster software and course materials for teaching numerical modelling in the environmental sciences. Transactions in GIS, 5, pp. 99-110.

Karssenberg, D., K. de Jong & P.A. Burrough (2000), A prototype computer language for environmental modeling in the temporal, 3D spatial and stochastic dimension. In Proceedings of the 4th International Conference on Integrating GIS and Environmental Modeling, September 2-8, Banff, Canada. http://www.colorado.edu/research/cires/banff (Access date: 14/11/2000)

Kessel, G. (1999), Biological control of Botrytis spp. by Ulocladium atrum, an ecological analysis. Wageningen: Wageningen University (PhD Thesis Wageningen University, Wageningen, the Netherlands).

Leavesley, G.H., P.J. Restrepo, S.L. Markstrom, M. Dixon & L.G. Stannard (1998), The Modular Modelling System (MMS): User's Manual. U.S. Geological Survey. Open-File Report 96-151. Available at: http://wwwbrr.cr.usgs.gov/mms (Access date: 14/11/2000)

Li, R-M, D.B. Simons & M.A. Stevens (1975), Nonlinear kinematic wave approximation for water routing. Water Resources Research 11, pp. 245-252.

Mackay, D.S. & L.E. Ban (1997), Forest ecosystem processes at the watershed scale: dynamic coupling of distributed hydrology and canopy growth. Hydrological Processes 11, pp. 1197-1217.

MATLAB (2001), Info at: http://www.mathworks.com/ (Access date: 02/09/2001).

Merriam, R.A. (1973), Fog drip from artificial leaves in a fog wind tunnel. Water Resources Research 9, pp. 1591-1598.

ModelMaker (2001), info at: http://www.modelkinetix.com/modelmaker/ (Access date: 02/09/2001).

Morgan, R.P.C., J.N. Quinton, R.E. Smith, G. Govers, J.W.A. Poesen, K. Auerswald, G. Chisci, D. Torri & M.E. Styczen (1998), The european soil erosion model (EUROSEM): a dynamic approach for predicting sediment transport from fields and small catchments. Earth Surface Processes and Landforms 23, pp. 527-544.

Olivera, F. & D. Maidment (1999), Geographic information systems (GIS)-based spatially distributed model for runoff routing. Water Resources Research 35, pp. 1155-1164.

Olsthoorn, T.N. (1998), Groundwater Modelling: Calibration and the Use of Spreadsheets. Delft: Delft University Press.

Ousterhout, J.K. (1998), Scripting: Higher Level Programming for the 21st Century. IEEE Computer magazine, March 1998.

Pebesma, E.J., D. Karssenberg & K. de Jong (2000), The stochastic dimension in a dynamic GIS. In Proceedings of Compstat 2000, 14th Conference of the International Association for Statistical Computing, 21-25 August, Utrecht, the Netherlands.

PCRaster (2001), Information available at http://www.geog.uu.nl/pcraster (Access date: 02/09/2001).

Smith, R.E. & J-Y Parlange (1978), A parameter-efficient hydrologic infiltration model. Water Resources Research 14, pp. 533-538.

STELLA (2001), info at: http://www.hps-inc.com/edu/stella/stella.htm (Access date: 02/09/2001).

Simile (2001), A Modelling Environment for ecological and environmental modelling. Available at: http://www.ierm.ed.ac.uk/simile/index.html (Access date: 11/09/2001).

Takeyama, M. (1997), Building spatial models within GIS through Geo-Algebra. Transactions in GIS 2, pp. 245-256.

Tucker, G., N. Gasparini, R.L. Bras & S. Rybarczyk (1999), An object-oriented framework for distributed hydrologic and geomorphic modeling using triangulated irregular networks. In Proceedings of GeoComputation 99, 4th International GeoComputation Conference, Fredericksburg, USA, 25-28 July 1999. http://www.geovista.psu.edu/geocomp/geocomp99 (Access date: 14/11/2000)

Van der Perk, M. (1997), Effect of model structure on the accuracy and uncertainty of results from water quality models. Hydrological Processes, 11, pp. 227-239.

Van der Perk, M., T. Lev, A.G. Gillet, J.P. Absalom, P.A. Burrough, N.M.J. Crout, E.K. Garger, N. Semiochkina, Y.V. Stephanishin & G. Voigt (2000), Spatial modelling of transfer of long-lived radionuclides from soil to agricultural products in the Chernigov region, Ukraine. Ecological Modelling, 128, pp. 35-50.

Van Deursen, W.P.A. & J.C.J. Kwadijk (1993), RHINEFLOW: an integrated GIS water balance model for the river Rhine. In Applications of Geographic Information Systems in Hydrology and Water Resources Management, HydroGIS 1993, Kovar K and Nachtnebel HP (eds), IAHS Publication 211: 507-518.

Van Deursen, W.P.A & G.W. Heil (1993), Analysis of heathland dynamics using a spatial distributed GIS model. Scripta Geobotanica 21, pp. 17-28.

Van Deursen, W.P.A. (1995), Geographical Information Systems and Dynamic Models. Utrecht: Koninklijk Nederlands Aardrijkskundig Genootschap/Faculteit Ruimtelijke Wetenschappen, Universiteit Utrecht.

Van Deursen, W.P.A., C.G. Wesseling & D. Karssenberg (2000), How do we gain control over GIS technology? In Proceedings of the 4th International Conference on Integrating GIS and Environmental Modeling, September 2-8, Banff, Canada. http://www.colorado.edu/research/cires/banff (Access date: 14/11/2000)

Van Dijck, S.J.E. (2000), Effects of agricultural land use on surface runoff and erosion in a Mediterranean area. Utrecht: Koninklijk Nederlands Aardrijkskundig Genootschap/Faculteit Ruimtelijke Wetenschappen, Universiteit Utrecht.

Van Langevelde, F. (2000), Scale of habitat connectivity and colonization in fragmented nuthatch populations. Ecography 23, pp. 614-622.

Wesseling, C.G., D. Karssenberg, W.P.A van Deursen & P.A. Burrough (1996a), Integrating dynamic environmental models in GIS: the development of a Dynamic Modelling language. Transactions in GIS 1, pp. 40-48.

Wesseling, C.G., W.P.A van Deursen & P.A. Burrough (1996b), A spatial modelling language that unifies dynamic environmental models and GIS. In Proceedings, Third International Conference/Workshop on Integrating GIS and Environmental Modelling, Santa Fe, NM, January 21-26, 1996. Santa Barbara, CA: National Center for Geographic Information and Analysis. http://www.ncgia.ucsb.edu/conf/SANTA_FE_CD-ROM/main.html (Access date: 14/11/2000).

Woolisher, D.A., R.E. Smith & D.C. Goodrich (1990), KINEROS, a kinematic runoff and erosion model: documentation and user manual: 130

Zheng, C. (1990), MT3D, a Modular Three-Dimensional Transport Model for simulation of advection, dispersion and chemical reaction of contaminants in Groundwater Systems. The United States Environmental Protection Agency.

Zheng, C. (1993), Extension of the method of characteristics for simulation of solute transport in three dimensions. Ground water 31, pp. 456-465.

3. A PROTOTYPE DYNAMIC ENVIRONMENTAL MODELLING LANGUAGE FOR MODELLING IN TWO AND THREE SPATIAL DIMENSIONS

Derek Karssenberg (with Kor de Jong)

Abstract: Environmental modelling languages are programming languages developed as a tool for building computer models simulating environmental processes. They come with database and visualisation routines for the data used in the models. Environmental modelling languages provide the possibility to construct dynamic models, also called forward models, which are simulations run forward in time, where the state of the model at time *t* is defined as a function of its state in a time step preceding *t*. Nowadays, these modelling languages can deal with simulations in two spatial dimensions, but existing software does not support the construction of models in three dimensions. We describe concepts of an environmental modelling language supporting dynamic model construction in two and three spatial dimensions. The lateral dimension is represented by gridded maps, with a regular discretisation, while the vertical dimension is represented by an irregular discretisation in voxels. Universal spatial functions are described with these entities of the modelling language as input. Dynamic modelling through time is possible by combining these functions in structured script sections, providing a section which is executed repetitively, representing the time steps. The concepts of the language are illustrated with an example model, built with a prototype of the language.

3.1 Introduction

In the environmental sciences, numerical computer models are a powerful means to represent and communicate our understanding of the environment, and have a wide application in predicting future changes in natural processes, often as a result of human impact. Since most natural processes have spatial components that change through time, these environmental computer models are typically spatial, with a flow of information in two or three dimensional space, and over time. The temporal behaviour is mostly simulated by dynamic modelling, using rules of cause and effect in time. Dynamic modelling involves a simulation which is run forward in time, where the state of a model at time *t* is defined as a function of its state in a period or time step preceding *t*, mostly represented as *t*-1. The function representing the transition of the state of a model between *t*-1 and *t* can be a deterministic, empirical, or probabilistic function, or a set of interrelated functions. Examples of environmental models are found in research fields such as hydrology, ecology, crop science, soil science, sedimentology, climatology, glaciology.

Environmental models have been and still continue to be programmed using system programming languages such as C++ or Fortran, because most environmental modellers are used to building models this way, and also because these languages enable almost any model to be programmed. Although not widely used, modelling languages included in

many Geographical Information Systems (GIS) are a potential alternative for developing environmental models, since their concepts provide advantages over system programming languages, when applied for environmental model construction (Chapter 2). Unlike system programming languages, modelling languages in GIS

- provide pre-programmed functions in which commonly used spatial algorithms have been pre-programmed,
- provide these functions in a suitable way such that they can be glued together in a model by a modeller using his or her understanding of environmental processes rather than computer expertise,
- provide generic tools for database management and visualisation of data read and written by the model.

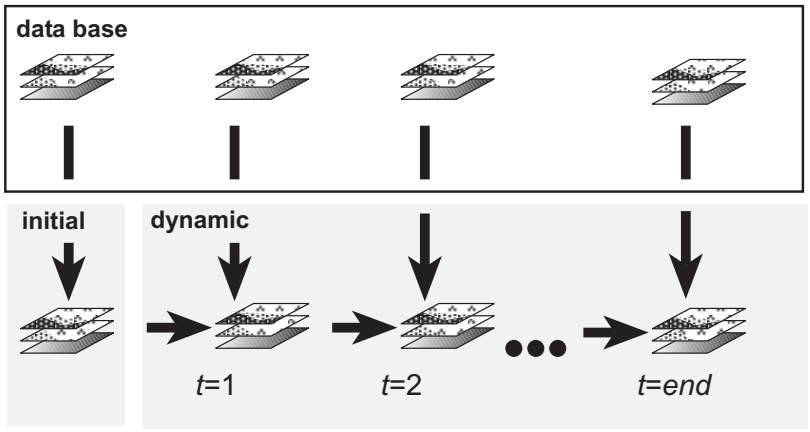
The reasons why standard GIS (e.g. ESRI 2001, IDRISI 2001) have so far failed to become important tools for environmental model construction is mainly because of their limited set of spatial functions and a lack of efficient interfaces for spatio-temporal model construction, such as scripting environments, to glue these together in temporal model scripts. In addition, most proprietary GIS lack visualisation tools and database management for temporal data, such as rain time series or time series of vegetation patterns. Even though standard GIS provide insufficient functionality for environmental model construction, the three conceptual advantages of GIS modelling languages over system programming languages, as noted above, still exist. This has driven specialist research groups to develop new modelling languages with sufficient functionality for spatio-temporal model building, embedded in GIS (e.g. Takeyama and Couclelis 1997, GRASS 2001, PCRaster 2001). For environmental model building, these so called environmental modelling languages now provide an alternative to system programming languages. For instance, the PCRaster language (Van Deursen 1995, Wesseling *et al.* 1996, PCRaster 2001) includes 120 spatial and non spatial functions on gridded maps. Their syntax follows mathematical notation:

$$ResultMap = \mathbf{function}(InputMap_{1..n}) \qquad (1)$$

where **function** is a pre-programmed function, *ResultMap* is its output and $InputMap_{1..n}$ are the input maps. For model building, these functions are glued together in a model script (Figure 3.1), structured in sections. The functions in the 'initial' section derive a set of maps $Map_{1..n}$ from base maps available in the database. Each map contains values representing a spatial variable used in the model, where the initial section results in attribute values representing the state of the model at the start of the model run, at t=0. The temporal behaviour is represented by the set of operations in the 'dynamic' section. This same set of operations calculates for each time step the attribute values on $Map_{1..n}$ for that time step. For each time step t+1, the operations in the dynamic section use the values on $Map_{1..n}$ at time step t as their input or maps read from the data base, for each time step.

This simple approach of gluing together spatial operations in a model script section which is repeatedly executed, has proved to be powerful for dynamic spatial

(A)



(B)

```
initial
 # sequence of spatial functions
 ...
 ...

dynamic (nrtimesteps = ...)
 # sequence of spatial functions
 ...
 ...
```

Figure 3.1. (A) Concepts and (B) modelling script for spatio-temporal modelling in GIS.

environmental model building in a wide range of fields, which is shown by the case studies described in Chapter 2, and Chapters 5-8, and many other research studies (e.g., De Roo *et al.* 2000, De Wit, 2001, Eleveld 1999, Van der Perk *et al.* 2001, Van Deursen and Heil 1993, Kessel 1999, Van Langevelde 2000). An evaluation of the PCRaster modelling software for hydrologic model building in Chapter 2 (published as Karssenberg 2002), showed that its successful, wide, application is mainly because the pre-programmed functions can easily be glued together in a model script, without the need of specialist programmers. But the same evaluation revealed many weaknesses of environmental modelling languages. The main disadvantage lies in the restricted functionality. In particular, the existing spatio-temporal environmental modelling languages do not provide functionality for dynamic modelling in three dimensions nor for handling error propagation in dynamic modelling. This chapter focuses on modelling in three dimensions, while chapter 4 will deal with error propagation modelling.

In addition to the two lateral dimensions, the third, vertical dimension, has an obvious relevance in many environmental models that are used to simulate forward in time. Three dimensional models are found in fields such as hydrology and soil science (e.g., groundwater flow modelling, Harbaugh and McDonald 1996), sedimentology (e.g., Chapter 7, published as Karssenberg *et al.* 2001), marine and lacustrine sciences (e.g., water circulation modelling, Mason *et al.* 1994), crop and vegetation science (e.g., canopy modelling, Song *et al.* 1997), and meteorology. These models are generally developed in system programming languages with a restricted application of GIS for

visualisation of model inputs and outputs. Although three dimensional GIS exist (e.g., EarthVision 2002, LYNX 2002), these are not used for dynamic modelling, since their functionality is focused on database management, static (non temporal) operations, such as volume modelling, and visualisation. There is a strong need to extend existing environmental modelling languages, such as the PCRaster language described above, which do provide functionality for dynamic modelling, with three-dimensional functionality, in addition to their two dimensional functions on maps.

This chapter describes how environmental modelling languages can be extended with functionality for three-dimensional modelling, and provides a prototype environmental modelling language encapsulating these concepts, which is partly developed using functions from the PCRaster software (Van Deursen 1995, PCRaster 2002). The outline is as follows. First, the application field of the developed language is described. Second, the concepts of the language itself, its entities and functions, and the syntax is described while a prototype language developed following these concepts is used in an example model to illustrate its application. Finally, the results are discussed.

3.2 Application field

3.2.1 Spatial dimension

The environmental modelling language is meant to simulate environmental processes occurring within a three dimensional block of material such as rock, air, and/or vegetation. The focus is on spatially continuous phenomena. Processes dealing with individuals moving in space, such as animals (e.g., Westervelt and Hopkins 1999, Bian 2000), is not dealt with, since this needs a different approach which is beyond the scope of this research. Within the three dimensional block, different units, such as rock layers, can be distinguished, each with their specific properties for the process simulated. The shape of the units and the block may be fixed or changing. It is fixed, when flow of material or information within the block is modelled (Figure 3.2A). Examples of such models are found in hydrology, oceanography, geochemistry, climatology, meteorology, pedology, ecology. For instance, in groundwater modelling (e.g., Harbaugh and McDonald 1996), flow is simulated in and between aquifers that mostly do not change in form and location in time spans represented by the model.

The shape of the block will change when flow of material is simulated resulting in a change in shape of existing units, removal, or addition of units (Figure 3.2B). Models in fields such as geomorphology (e.g., Ahnert, 1987), sedimentology (e.g., Tetzlaff and Harbaugh, 1989; Mackey and Bridge 1995), petroleum engineering (Chapter 7, published as Karssenberg *et al*. 2001), glaciology, use a block of solid earth or ice where erosion and deposition of material causes addition or abstraction of layers of material on the top side of the block. Processes such as compaction, faulting, soil creep, will cause a change in shape of the units inside the block. In plant growth models, the block of units could represent a canopy, and growth of new layers of plants could extend the block with new plant layers on the top side. Our focus is on changes in the shape of the block by movement, deposition, or compression in vertical direction, ignoring changes in the block in lateral direction, such as caused by folding.

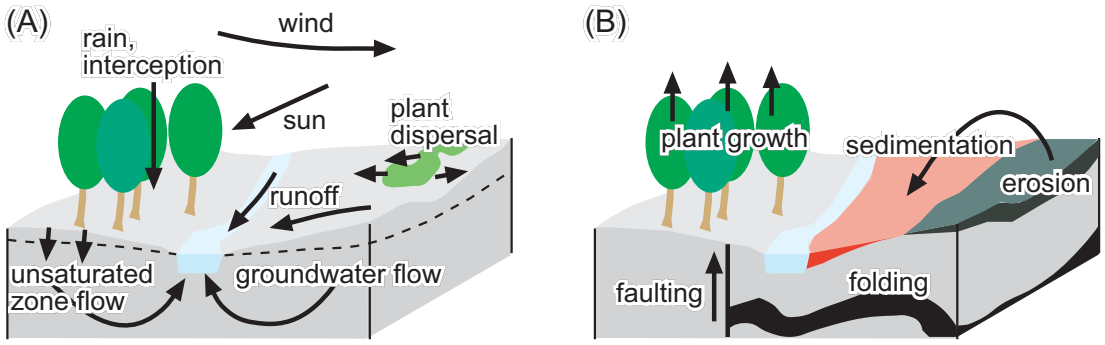


Figure 3.2. (A) Processes in a three-dimensional block with a constant shape, (B) processes changing the shape of a three dimensional block.

3.2.2 Temporal dimension

The focus is on process simulating environmental models that use the rules of cause and effect forward in time, with a discretisation of time in time steps. The state of the model variables, which are attributes in one, two, or three dimensional space, at time t+1 is defined by their state at t and a function f, with associated parameters $P_{1..k}$:

$$A_{1..m}(t+1) = \mathrm{f}(A_{1..m}(t), B_{1..n}(t), t, P_{1..k}) \qquad (2)$$

The model variable(s) $A_{1..m}$ belong to coupled processes, and therefore have feedback in time, for instance stream water level. The model variable(s) $B_{1..n}$ for all time steps and $A_{1..m}$ at t=0 are simple inputs to the model, for instance incident rainfall in a runoff model and initial plant distribution, respectively. Without $B_{1..n}$ the model would represent a closed system. The function f models the change in the state of all model variables over the time step t to t+1, and it mostly represents a numerical solution of a set of differential equations, such as groundwater flow equations or plant growth equations. Alternatively, or additionally, the function f may represent transitions in the state of the model simulated as empirical functions, an abrupt transition, or cellular automata (e.g., Miyamoto and Sasaki 1997). It may also include probabilistic rules when model behaviour is better described as a stochastic process, which will be covered by Chapter 4.

3.3 Entities of the language

3.3.1 Introduction

Entities are the basic objects that carry the data which are changed by the modelling functions. Although the function f in equation 2 principally operates at all locations in three dimensional space, some processes typically represent only the flow of material or information in the lateral direction, while other processes are truly three dimensional, such as groundwater flow. For this reason, two entities are used in the language: two

dimensional ‘maps’, with a spatial discretisation in ‘cells’, and three dimensional ‘blocks’, with a spatial discretisation in ‘voxels’, see Figure 3.3. Both entities use the same discretisation of the temporal dimension. In addition, the discretisation of the lateral spatial dimension is the same for maps and blocks. This correspondence in discretisation between maps and blocks guarantees efficient exchange of information between the entities, in the same modelling environment.

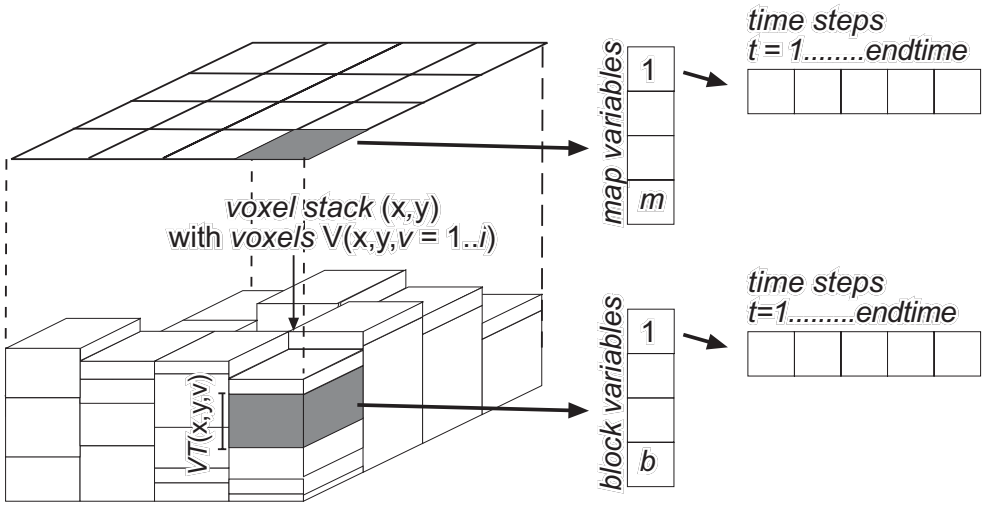


Figure 3.3. Entities of the language. Left, map and block; centre, list of map and block variables; right, one dimensional matrix with values for each time step, stored for each map variable, for each cell or voxel.

3.3.2 Maps

A map discretises the lateral space with a regular grid, as applied in many GIS’s. Other possible approaches would be a vector approach or an irregular grid discretisation but their advantages do not compensate for the advantages of a regular grid discretisation. Advantages of a regular grid are 1) its fixed neighbourhood for each cell, which makes numerical solution schemes more straightforward, 2) its constant ‘support’ of one grid cell, implying that cell values and functions used in a model represent the average state or behaviour in an area of a constant cell size, 3) the availability of many numerical solution schemes, including cellular automata. A vector approach would be needed for change of shape in lateral direction (Raper 2000), caused by for instance folding inside the three dimensional block, but these processes are not meant to be modelled with the language.

Each raster cell on a map has a spatial location (x,y), and contains the same map variables $M_{1..m}$ (Figure 3.3). For each map variable, the temporal dimension is represented by a one dimensional array which is filled with cell values during a model run. At the end of a model run, each field in this array contains an attribute value for time step t, for the cell at (x,y). Each map variable is assigned a data type (Boolean, nominal, ordinal, directional or local drain direction). Advantages of such a data typing mechanism for spatial data in a raster modelling language were described by Van Deursen (1995) and Wesseling *et al*. (1996).

3.3.3 Blocks

A block discretises the lateral direction with the same regular grid of the maps used in a model, providing a tight link between maps and blocks. For representing the vertical direction in a block, there are at least three different approaches to choose from. The first approach shown in Figure 3.4A, using a regular discretisation of the vertical direction, has the advantage that each voxel has the same, fixed, set of neighbouring cells, which provides a straightforward framework for implementing spatial operations. In spite of this advantage, this approach is not used here, because, 1) in often occurring cases of units with a very small thickness, such as thin depositional units, it would require a small voxel thickness in the whole block to represent these units, resulting in an unacceptably large number of voxels in the block, 2) the fixed vertical thickness of voxels would result in unacceptably large numerical errors when a certain amount of volume (e.g. sediment) is added on the top of the block, or in the case of a decrease in thickness of existing units in the block (e.g. compaction).

The second approach (Figure 3.4B), with a fixed number of voxels at each x,y location, each with a different thickness, is very efficient for representing laterally continuous layers. As such, it is widely used in groundwater flow modelling (e.g. Harbaugh and McDonald 1996) or meteorological modelling, but it is inadequate when layers are discontinuous, for instance in the case of complex geological formations. The third approach with a variable voxel thickness and a variable number of voxels per x,y location (Figure 3.3) is used here. It assumes a strong relation between the discretisation and the thickness of units, such as volumes with a specific rock type, or trees with a certain property (see Figure 3.1). Preferably, the thickness of a voxel corresponds with the thickness of a certain unit at that location, although exceptions may occur, when multiple voxels on top of each other are used to represent the same unit. Note that the discretisation in vertical direction is analogue to a vector (polygon) representation of different units in two dimensions, for instance applied on a soil type map, where the boundary between polygons is defined by an abrupt change in a certain property. The same holds for boundaries between voxels in the vertical direction, applied here. Advantages of the approach used here are 1) an exact representation of the vertical position and thickness of units, 2) the possibility to add volumes of material or to change

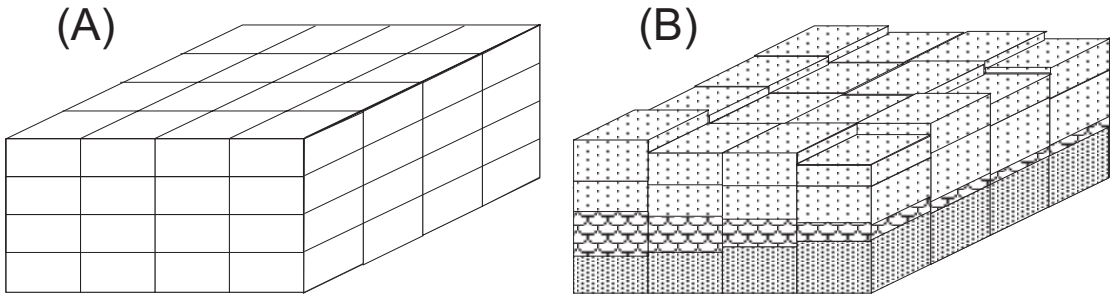


Figure 3.4. Discretisation of the vertical dimension, A) regular discretisation, and B) fixed number of voxels at each x,y location, each with a different thickness.

thickness of units inside the block within acceptable ranges of error, 3) minimization of the number of voxels, since each unit is represented by one voxel, independent of the thickness of the unit, 4) generality, since the discretisations described in the first two approaches above are subsets of this discretisation. The main disadvantage of this approach is that the number and location of neighbouring voxels will be different for each voxel. As a result, spatial operations requiring a topology in three dimensions will be complicated compared to spatial operations on a map, with a regular discretisation. But this disadvantage is considered of minor importance, mainly because the use of operations requiring a topology in three dimensions is expected to be less frequent than the use of other operations, such as recoding voxel values or operations requiring just the topology in vertical direction (e.g. vertical fluid flow), which *is* constant for all voxels.

In the approach applied here, each location (x,y) on a regular grid contains a stack of voxels with a temporally variable number of i voxels $V(x,y,v)$, with $v=1\ldots i_{x,y}$ (Figure 3.3). Each voxel $V(x,y,v)$ has its own temporally variable voxel thickness $VT(x,y,v)$ and the bottom elevation of each voxel stack is the voxel stack bottom $VB(x,y)$. The top elevation of each voxel stack is $VB(x,y)$ plus the sum of $VT(x,y,v)$ over $v=1..i_{x,y}$. Each voxel contains the same block variables $B_{1..b}$ with an array representation of attribute values in the temporal dimension also used for map variables.

3.4 Functions of the language

3.4.1 General concepts

The function f (eq. 2) is represented by one or several functions on the map or block entities of the modelling language. Functions are chosen representing universal functions on spatial entities, which can be applied in a wide range of models. Many operations make sense in both the two dimensional and three dimensional domain, think for instance of distance calculation. For this reason, most functions work both on map and block entities, and the operation is modified according to the input entity. For instance, a distance calculation function on a map results in distances in two dimensions, given on a map, while the same calculation on a block would result in a block entity with distances. This polymorphic behaviour cannot be used for all functions, and a small set of functions is provided that work only on either maps or blocks. All functions check the data type of input entities, and might change their behaviour according to the data type of the inputs (Van Deursen 1995, Wesseling *et al.* 1996).

One group of functions results in a change of cell or voxel values on a map or in a block, respectively. These are referred to as ‘functions on maps and blocks, no change of form in spatial dimension’. In addition, a group of functions may change the thickness of voxels or add or remove voxels from a block. These are referred to as ‘functions on blocks, change of form in spatial dimension’. This group of functions is not used for map entities, since the size and number of cells on a map is constant.

3.4.2 Functions on maps and blocks, no change of form in spatial dimension

1) point functions
The point functions derive attribute value(s) of a cell or voxel from one or more attribute value of the cell or voxel itself only (Figure 3.5A). Examples are mathematical functions such as addition, subtraction, logical selection on attributes of the cell or voxel, or functions returning the thickness of a voxel.

2) direct neighbourhood functions
The direct neighbourhood functions derive attribute value(s) of a cell or voxel from attribute value(s) in a spatially restricted neighbourhood. Examples are two- or three-dimensional moving filters, computing a new value of the centre cell or voxel of a 2D or 3D window as a function of attribute values in the window. An example of a direct neighbourhood function provided for blocks only, is a function assigning for each voxel an attribute value of the voxel immediately above it (Figure 3.5B, right), which could be used for simulating infiltration.

3) entire neighbourhood functions
The entire neighbourhood functions derive attribute value(s) of a cell or voxel from attribute values(s) in all cells on a map or block (Figure 3.5C). All functions involving the solution of flow equations in two or three dimensions belong to this group, e.g. groundwater flow. Other functions belonging to this group are calculators of Euclidean or relative distances to specific voxels.

4) functions with a neighbourhood defined by a given topology
This group of functions derives for each cell or voxel an attribute value from attribute values in a neighbourhood defined by an explicitly given topology. This topology defines connections between cells or voxels. An example of such a topology is a local drain direction network, representing flow directions over a map (Figure 3.5D). Functions using such a topology calculate catchment characteristics such as catchment area or slope length, or transport material in downstream direction over the network (Van Deursen 1995).

3.4.3 Functions on blocks, change of form in spatial dimension

These are functions that change the form of the block as a whole, but only in the vertical direction. This implies that the neighbourhood in the lateral direction around voxels changes. The following groups are distinguished.

1) top side block form functions
The top side block form functions remove or add material at the top of the block, in vertical direction. In the case of addition of material (Figure 3.6A), the attribute values of the volume added are specified. Addition is done in two ways. If the added volume has attribute properties similar to these of the voxel at the top of the block, the voxel thickness of this existing voxel is increased. If the added volume is different from the top

voxel regarding attribute properties, a new voxel is added to represent the addition. Removing volume means that voxels are completely or partially removed at the top side of the block (Figure 3.6B).

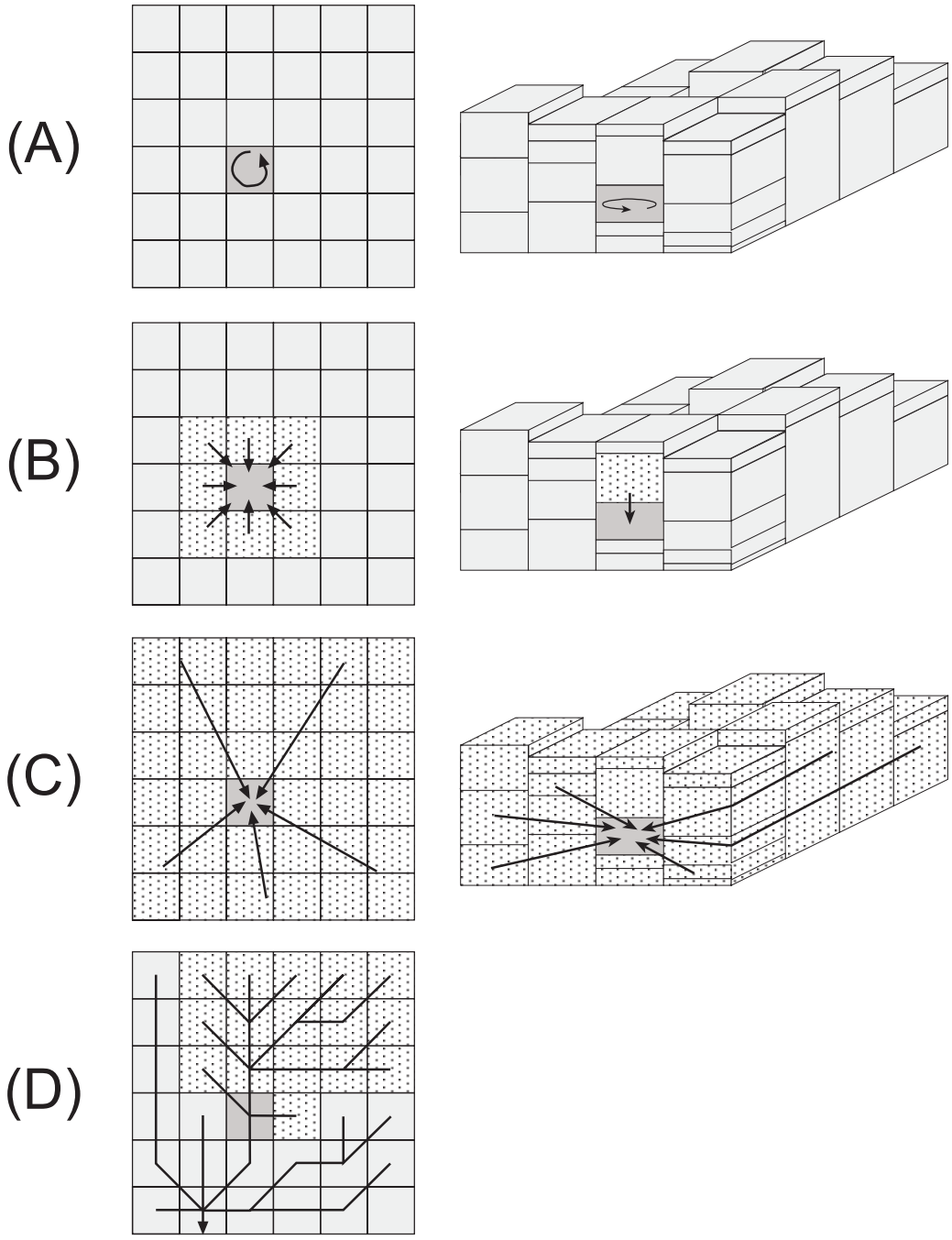


Figure 3.5. A) point functions, B) direct neighbourhood functions, C) entire neighbourhood functions, and D) functions with a neighbourhood defined by a given topology.

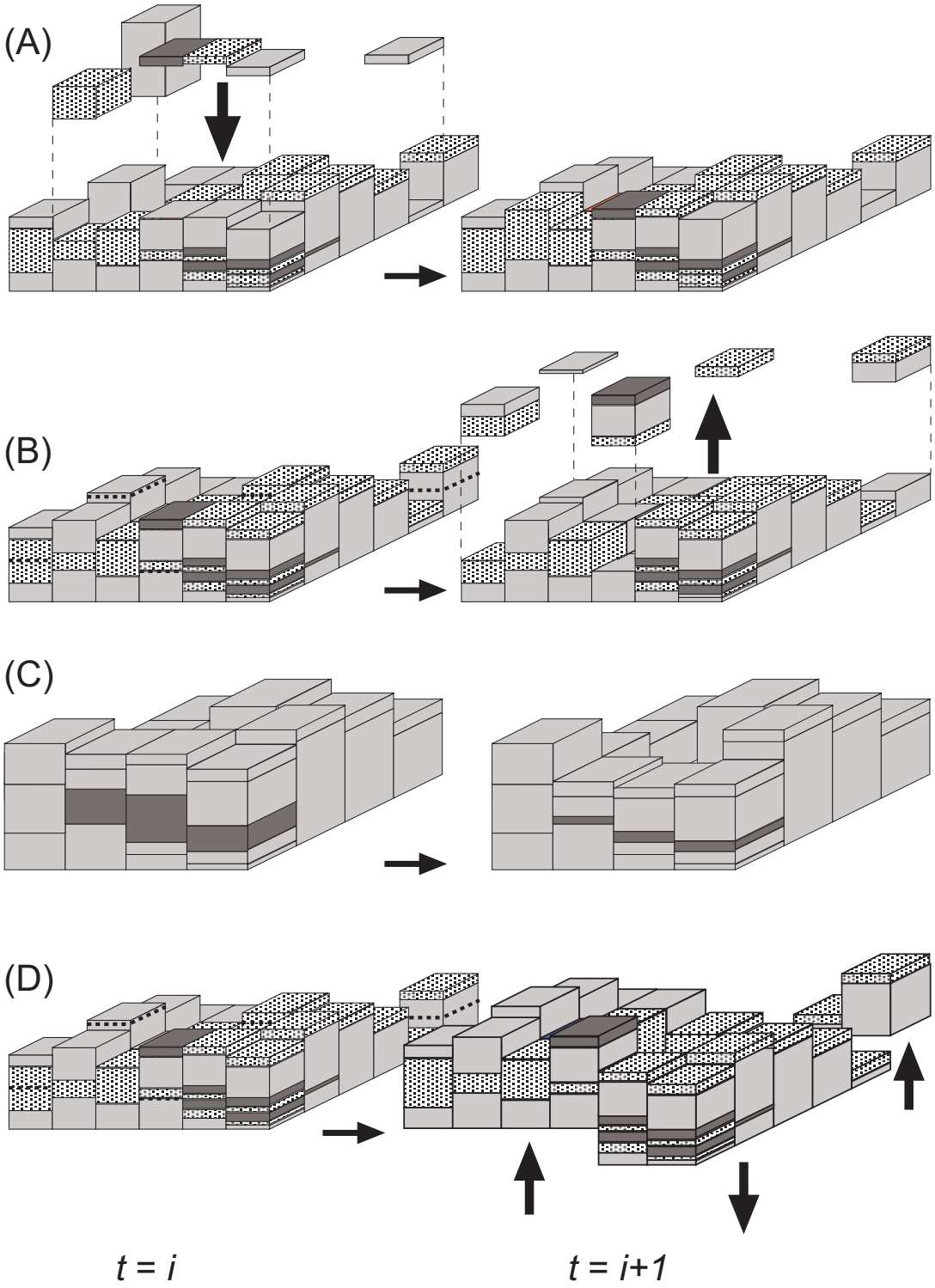


Figure 3.6. Functions on blocks, change of form in spatial dimension. A) top side block form functions, addition of material, B) top side block form functions, removal of material, C) inside block form functions, D) bottom side block form functions.

2) inside block form functions
The inside block form functions change the thickness of existing material inside the block, in vertical direction (Figure 3.6C). This is done by changing the thickness of existing voxels inside the block, without changing their attribute values. A change of thickness can be caused by 1) change of thickness of volume that is already inside the voxel itself, e.g. compaction or decompaction of rock in situ, or compression of air, 2) addition or abstraction of volume from the voxel as a result of flow of material between neighbouring voxels in lateral direction, e.g. soil creep.

3) bottom side block form functions
The bottom side block form functions change the location of volume in the block per x,y location, in a vertical direction, as a result of change of elevation of the bottom of the block (Figure 3.6D). If the block represents the subsurface, it could be caused by faults in the block.

4) resampling functions
A resampling to another discretisation in the vertical dimension is needed 1) when a regular discretisation is needed for solving a numerical algorithm in another function, e.g., for groundwater flow, while the available data for that algorithm are available in a block with an irregular discretisation, or 2) when two blocks with a different discretisation are combined in a function that needs blocks with a corresponding discretisation as input, for instance when spatial statistics are calculated for different realisations of a three dimensional block.

3.5 Syntax

3.5.1 Introduction

The framework provided by the language for invoking and combining operations in a model is mainly important for the activity of model building. While building a model, the modeller is forced to work in this framework and his or her way of thinking will be influenced by it. Existing environmental modelling languages use either a framework with a graphical representation (e.g., MODELMAKER 2002, STELLA 2002, ESRI 2002) or one with a written representation of a model, being a program or script (e.g. PCRaster 2002, GRASS 2002, ESRI 2002). Since the graphical representation represents functions, and their inputs or outputs as graphical entities, it has the advantage that it is very easy to use for inexperienced users. While building a model, the researcher simply connects these entities on the screen, resulting in a flow diagram similar to graphical representations of models often used in scientific reports (Forrester 1968). In spite of this advantage of graphical modelling languages, a written representation is proposed here. This is mainly because a written representation has the advantage of an exact definition of the model in a model script, while the functionality of a model represented by a flow diagram of a graphical representation is not always unambiguous. Apart from this, experienced modellers are not expected to prefer a graphical above a written representation of a

model, since they are familiar with the mathematical notation which is often applied by written modelling languages.

The syntax of the language should follow common concepts of mathematical notation and reasoning applied in scientific environmental modelling. The natural language approach to modelling which was used by Tomlin (1990) is convenient for simple spatial problems, but it cannot be used for more complex modelling. The operations of the language are better represented as functions with one or more inputs and one output, which can be nested. This results in programs or scripts of a model which look similar to a theoretical, mathematical description of the model, enhancing readability of the model code. In addition, the language should be understandable and easy to use for environmental model builders, who do not necessarily need to be experienced programmers. Technical details such as read/write definitions and storage allocation, should be avoided as much as possible, since they distract the modeller from constructing the model.

3.5.2 Syntax of functions

The syntax of the functions follows algebraic notation (like in Wesseling *et al.* 1996):

$$Result = \mathbf{function}(Input_{1..n})$$

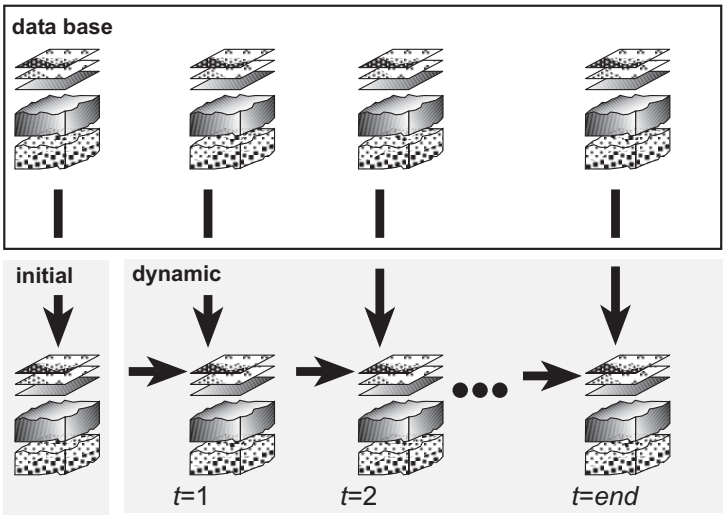
with one of the functions of the language, **function**, having the input map or block variables $Input_{1..n}$, resulting in the output map or block variable *Result*. In most cases, all input and output variables are either maps or blocks, although exceptions occur, for instance when an input map adds information to a block, in which case $Input_{1..n}$ are a block and a map, while the output is a block. The functions which make sense both in two and three dimensions show polymorphic behaviour, which means that their behaviour is adapted to the input(s). For instance, a point function **sqrt** with a map variable as input calculates the square root for each cell on a map, giving a map as *Result*, while the same function on a block would result in square roots for each voxel in a block. In addition, all functions check the data type of their inputs, and might change their behaviour depending on the data type of the input variable (c.f., van Deursen, 1995). The functions can be nested in a statement, where the result of one or more functions $\mathbf{function}_{1..l}$ is the input variable to another function:

$$Result = \mathbf{function}(\mathbf{function}_{1..l}(Input_{1..m}), Input_{1..n})$$

3.5.3 Script structure

The functions are combined in a script or program, structured in sections, see Figure 3.7. The concept of the script is similar to the concept of the script for spatio-temporal

(A)



(B)

```
initial
 # sequence of spatial functions
 ...
 ...

dynamic (nrtimesteps = ...)
 # sequence of spatial functions
 ...
 ...
```

Figure 3.7. (A) Concepts and (B) modelling script for temporal, two- and three-dimensional modelling in GIS.

modelling in two dimensions (Figure 3.1), although both maps and blocks can be used now. Each section contains a list of operations which are sequentially executed. The functions in the *initial* section are executed only once, at the start of the model run. These read data from the data base, generating map or block variables for the first time step. The *dynamic* section, representing the temporal change, is a section which is run for each time step. The functions in the dynamic section represent the change in the state of model variables over one time step (eq. 2).

3.6 3D spatial and temporal example model

A prototype implementation of the modelling language proposed here has been created by extending an existing scripting language (Python, 2001) with our 3D modelling domain specific data structures (maps and blocks) and functions. The data structures and algorithms were written in a system programming language (C++), using existing code from the PCRaster modelling language (Van Deursen 1995, PCRaster, 2001). An example model was made predicting the three dimensional architecture of river deposits, simulating the formation of a series of deposits formed by the river channel, with

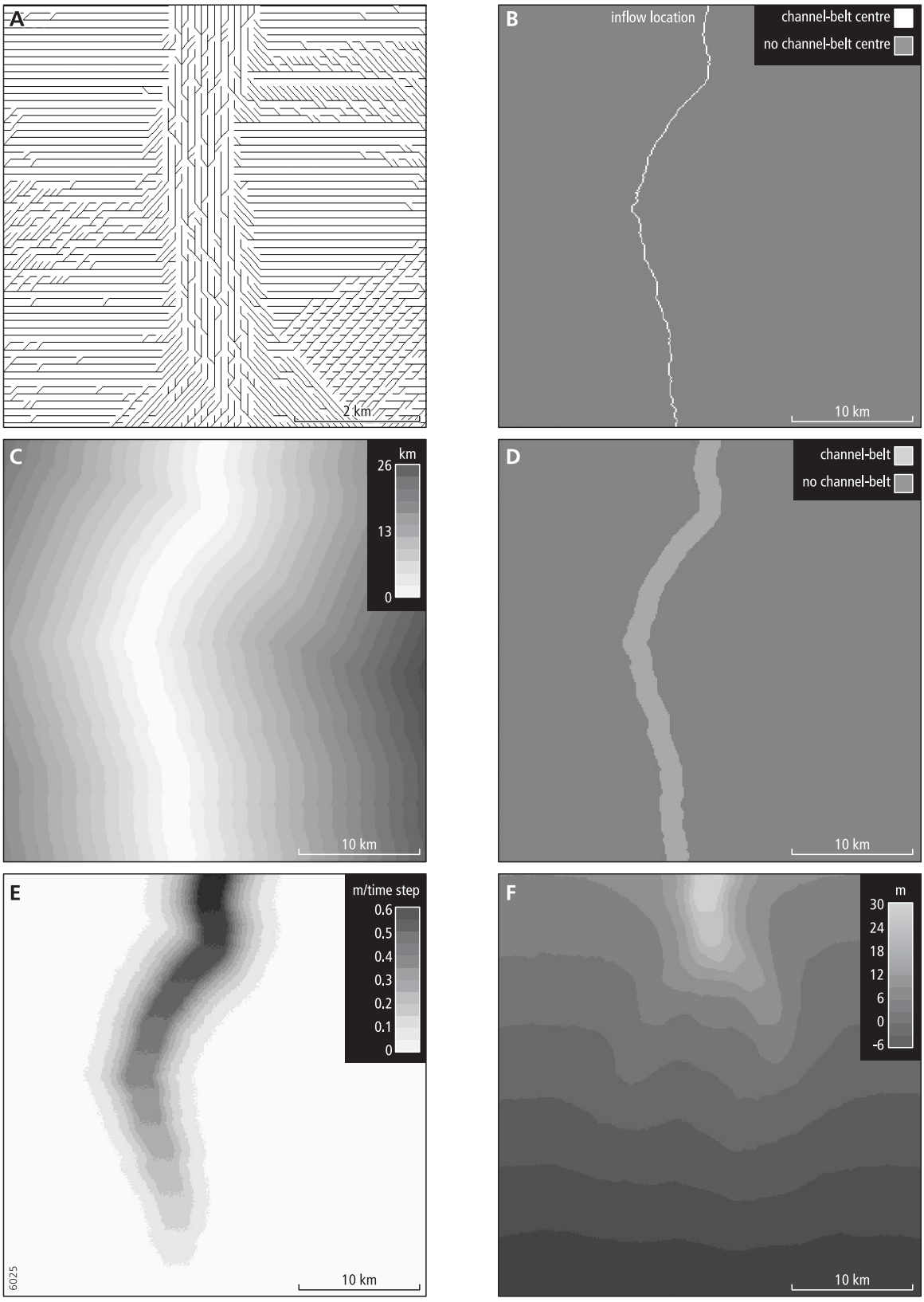
associated overbank deposits next to these channel belt deposits. Table 3.1 gives the script for a simplified version of the model described by Mackey and Bridge (1995) and in Chapter 7 (published as Karssenberg *et al.* 2001). Figure 3.8 gives the set of maps created for one time step in the model, representing the formation of one channel belt, and its associated deposits.

For each of the 20 time steps, first, a local drain direction map (`Ldd`) is derived from the topographical elevation at that time step (`Dem`), using the eight point pour algorithm implemented in the **lddcreate** function (Van Deursen 1995; Burrough and McDonnell 1998). The **path** operator calculates the centre of the channel belt (`Centre`) by following the downstream path over `Ldd`, starting at the channel inflow location at the top of the map, which is the map `In`. The channel belt (`Belt`) consists of all cells with a distance less than half the width of the channel belt, which is 750 m in this case. This map

```
initial
  Depos=depos.map        # deposition rate at channel belt
                         # (m/timestep)
  In=inflow.map          # inflow location
  Dem=dem.map            # initial elevation model

  dynamic (nrtimesteps=20)
    # create a local drain direction map
    Ldd=lddcreate(Dem)
    # centre of channel belt, path downstream from the inflow
    # point
    Centre=path(Ldd,In)
    # distance to the channel belt centre line (m)
    Dist=spread(Centre,0,1)
    # channel belt with width of 1500 m
    Belt=Dist < 750
    # deposition (m/timestep)
    Add=if(Belt then Depos else (Depos*exp((750-Dist)/1500)))

    Erosion=if(Belt then 10 else 0)
    Deposition=if(Belt then 10+Add else Add)

    Lith=remove(Erosion)
    Lith=add(Deposition, Belt)

    Dem=top()
```

Table 3.1. Alluvial architecture modelling script, remarks are behind a #.

Figure 3.8 (next page). Alluvial architecture model, one realization at time step 20, flow direction is from top to bottom on the maps. (A) zoomed area of local drain direction map (`Ldd` in Table 3.1), (B) channel belt centre cells (`Centre`), (C) distance to the channel belt centre (`Dist`), (D) channel belt (`Belt`), (E) deposition (`Add`, m/time step), and (F), topographical elevation (`Dem`, m) after erosion and deposition, used for the next time step.


A
2 km
B
inflow location
channel-belt centre
no channel-belt centre
10 km
C
km
26
13
0
10 km
D
channel-belt
no channel-belt
10 km
E
m/time step
0.6
0.5
0.4
0.3
0.2
0.1
0
10 km
6025
F
m
30
24
18
12
6
0
-6
10 km

is calculated by the two **spread** operations, which calculate the distance from the TRUE cells on `Centre`, resulting in the map `Dist`, and a 'less than' (**<**) function operating on `Dist`. The pattern of deposition (`Add`) is derived from the channel belt map, assuming a negative exponential decrease of deposition with distance from the channel belt. The variable `Lith` is the three dimensional block with the lithology, containing channel belt deposits in a matrix of overbank deposits. This block is for each time step updated by the **remove** function, resulting in incision in the old deposits at the location of the channel belt, and the **add** function, resulting in filling of this incised band with channel belt deposits, and additional deposition next to the channel. Figure 3.9 gives a three dimensional output of one realisation of the model. Since deposition decreases with downstream distance, each new channel belt diverges from the previous one at the inflow location, which is called nodal avulsion (Mackey and Bridge, 1995).

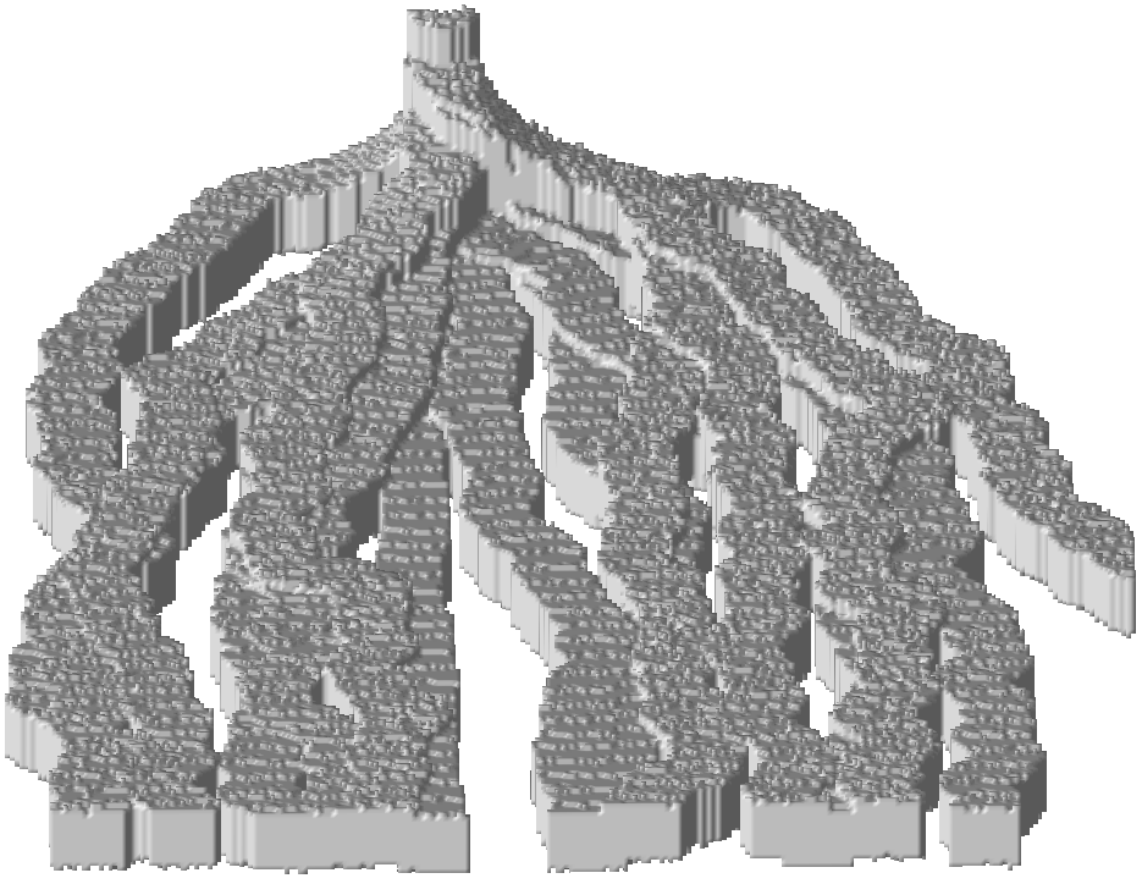


Figure 3.9. Example output of alluvial architecture model, timestep 20, three-dimensional picture of channel belt deposits, `Lith` in Table 3.3. Inflow point at top of image, lateral extension corresponds to Figure 3.8, thickness of individual channel belts is 10 m.

3.7 Discussion and conclusions

The goal of this study is to provide new concepts to extent existing dynamic spatial environmental modelling languages with functionality for modelling in three spatial dimensions. It was possible to implement an example model with the prototype language, built according to these concepts, while Chapter 7 provides a more complicated example. This does not mean that we have arrived at a final version of the language. Many steps need to be taken before a new grown up tool for construction of multi-dimensional models will be available. Below, a short discussion is given of the main weaknesses of the current prototype language.

Compared to two dimensional static or dynamic models, the amount of data related to the type of models worked with here, is enormous, since three spatial dimensions need to be represented. This is a problem both from the data storage point of view, but also with regard to the run times of models. Since this chapter focuses on the concepts of the language and its entities, these problems are not dealt with in the prototype language. So, optimisation routines need to be developed. Regarding data storage, the main optimisation possible is to reduce the number of voxels in three dimensional blocks until acceptable degrees of resolution, by built-in resampling. This would also decrease run times of the models, since the data volume decreases.

In order to be useful for environmental researchers, the language needs to match their way of thinking. It should be possible for these people to understand the concepts intuitively. Although different approaches are possible, the approach regarding the script structure and syntax of the functions used here is similar to concepts of other existing environmental modelling languages (e.g., PCRaster, 2002). Since these languages have a wide application, it is expected that researchers will understand the concepts of the language proposed here, too.

Acknowledgements
We wish to thank Peter Burrough, Cees Wesseling, Willem van Deursen, Marc Bierkens, and Edzer Pebesma for giving many inputs to the research described.

3.8 References

Ahnert, F. (1987), Process-response models of denudation at different spatial scales. Catena Supplement 10, pp. 31-50.

Bian, L. (2000), Object-oriented representation for modelling mobile objects in an aquatic environment. International Journal of Geographical Information Systems 14, pp. 603-623.

Burrough, P.A., R.A. McDonnell (1998), Principles of Geographical Information Systems. Oxford: Oxford University Press.

De Roo, A.P.J., C.G. Wesseling & W.P.A. van Deursen (2000), Physically based river basin modelling within a GIS; the LISFLOOD model. Hydrological Processes 14, pp. 1981-1992.

De Wit, M. (2001), Nutrient fluxes at the river basin scale. I: the PolFlow model. Hydrological Processes 15, pp. 743-759.

EARTHVISION (2002). EarthVision, Dynamic Graphics, Inc., info at: http://www.dgi.com

Eleveld, M. (1999), Exploring coastal morphodynamics of Ameland (the Netherlands) with remote sensing monitoring techniques and dynamic modelling in GIS. Amsterdam: Universiteit van Amsterdam (PhD Thesis).

ESRI (2002), Environmental Systems Research Institute. Info at: http://www.esri.com/

Forrester, J.W. (1968), Principles of systems. Text and workbook chapters 1 through 10. Cambridge, Massachusetts USA: Wrigh-Allen Press, Inc.

GRASS (2002), info at http://www.geog.uni-hannover.de/grass/

Harbaugh, A.W. & M.G. McDonald (1996), User's documentation for MODFLOW-96, an update to the U.S. Geological Survey Modular Finite-Difference Ground-Water Flow Model. U.S. Geological Survey.

IDRISI (2001), info at http://www.clarklabs.org/

Karssenberg, D. (2002), The value of environmental modelling languages for building distributed hydrological models. Hydrological Processes, in press.

Karssenberg, D., T. Törnqvist & J.S. Bridge (2001), Conditioning a process-based model of sedimentary architecture to well data. Journal of Sedimentary Research 71, pp. 868-879.

Kessel, G. (1999), Biological control of Botrytis spp. by Ulocladium atrum, an ecological analysis. Wageningen, the Netherlands: Wageningen University.

LYNX (2002), Lynx Geosystems, 2001, info at http://www.lynxgeo.com

Mackey, S.D. & J.S. Bridge (1995), Three-dimensional model of alluvial stratigraphy: theory and application. Journal of Sedimentary Research B65, pp. 7-31.

Mason, D.C., A. O'Connall & S.B.M. Bell (1994), Handling four-dimensional geo-referenced data in environmental GIS. International Journal of Geographical Information Systems 8, pp. 191-215.

Miyamoto, H. & S. Sasaki (1997), Simulating lava flows by an improved cellular automata method. Computers & Geosciences 23, pp. 283-292

MODELMAKER (2002), info at: http://www.modelkinetix.com/modelmaker/

PCRaster (2002), info software and manuals at http://www.geog.uu.nl/pcraster

PYTHON (2002), Python language website at http://www.python.org

Raper, J. (2000), Multidimensional Geographic Information Science. London: Taylor & Francis.

Song, B., J. Chen, P.V. Desanker, D.D. Reed, G.A. Bradshaw & J.F. Franklin, J.F. (1997), Modeling canopy structure and heterogeneity across scales: from crowns to canopy. Forest Ecology and Management 96, pp. 217-229.

STELLA. (2001), info at: http://www.hps-inc.com/edu/stella/stella.htm

Takeyama, M. & H. Couclelis (1997), Map dynamics: integrating cellular automata and GIS through Geo-Algebra. International Journal of Geographical Information Science 11, pp. 73-91.

Tetzlaff, D.M. & J.W. Harbaugh (1989), Simulating Clastic Sedimentation. New York: Van Nostrand Reinhold.

Tomlin, C.D. (1990), Geographic Information Systems and Cartographic Modelling. Englewood Cliffs, New Jersey, USA: Prentice Hall.

Van Der Perk, M., J.R. Burema, P.A. Burrough, A.G. Gillet & M.B. van der Meer (2001), A GIS-based environmental decision support system to assess the transfer of long-lived radiocaesium through food chains in areas contaminated by the Chernobyl accident. International Journal of Geographical Information Science 15, pp. 43-64.

Van Deursen, W.P.A. (1995), Geographical Information Systems and Dynamic Models. Utrecht: Koninklijk Nederlands Aardrijkskundig Genootschap/Faculteit Ruimtelijke Wetenschappen, Universiteit Utrecht.

Van Deursen, W.P.A. & G.W. Heil (1993), Analysis of heathland dynamics using a spatial distributed GIS model. Scripta Geobotanica 21, pp. 17-28.

Van Langevelde, F. (2000), Scale of habitat connectivity and colonization in fragmented nuthatch populations. Ecography 23, pp. 614-622.
Wesseling, C.G., D. Karssenberg, W.P.A. van Deursen & P.A. Burrough (1996), Integrating dynamic environmental models in GIS: the development of a Dynamic Modelling language. Transactions in GIS 1, pp. 40-48.
Westervelt, J.D. & L.D. Hopkins (1999), Modeling mobile individuals in dynamic landscapes. International Journal of Geographical Information Systems 13, pp. 191-208.

4 ADDING FUNCTIONALITY FOR MODELLING ERROR PROPAGATION IN A DYNAMIC, 3D SPATIAL ENVIRONMENTAL MODELLING LANGUAGE

Derek Karssenberg (with Kor de Jong)

Abstract: Environmental modelling languages provide the possibility to construct models in two or three spatial dimensions. These models can be static models, without a time component, or dynamic models. Dynamic models are simulations run forward in time, where the state of the model at time *t* is defined as a function of its state in a period or time step preceding *t*. Since inputs and parameters of environmental models are associated with errors, environmental modelling languages need to provide standard techniques to calculate how these errors propagate to the output(s) of the model. As these techniques are not yet available, this research describes concepts for extending an environmental modelling language with functionality for error propagation modelling. The approach models errors in inputs and parameters as stochastic variables, while the error in the model outputs is approximated with Monte Carlo simulation. A modelling language is proposed which combines standard functions in a structured script (program) for building environmental models, and calculation of error propagation in these models. A prototype implementation of the language is used in three example models to illustrate the concepts.

4.1 Introduction

Chapter 3 showed that it is possible to extend existing two-dimensional dynamic environmental modelling languages with an additional spatial dimension, resulting in a tool for dynamic model construction in two and three spatial dimensions. The language described in Chapter 3 is restricted to the case when inputs, parameters, and the model structure are exactly known. In many situations of modelling this is not the case, because of for instance measurement errors, or insufficient field data. Also, it may be difficult to identify the model structure, since many different model structures can be used to represent a process. For these cases, it is said that the inputs, parameters, and the model structure are associated with uncertainty or errors, and it is important to know the effect of this uncertainty or errors on the output of the model.

Error propagation modelling, also referred to as uncertainty analysis, allows the researcher to assess the error in the model output variables resulting from propagation of model input errors through the model, or resulting from uncertainty in the model structure with its associated parameters (Heuvelink 1998, Crosetto and Tarantola, 2001). It is an important issue in environmental modelling, since input data of environmental models have many different sources, with a wide range of errors associated with them (Thapa and Bossler, 1992). The propagation of these errors through complicated dynamic spatial environmental models can only be assessed with powerful computational techniques. Most of these techniques model errors as stochastic variables (e.g., Heuvelink, 1998).

Analytical solutions for error propagation as a result of spatial functions on stochastic variables exist, but these are only available for a limited number of relatively simple functions. As a result, error propagation in dynamic spatial environmental models is mostly calculated by Monte Carlo simulation modelling (Heuvelink, 1998). Although error propagation modelling has long been a subject of concern in the GIS research community, a framework to implement it, let alone the implementation itself, in a dynamic spatial environmental modelling language such as described in Chapter 3, is not yet provided. Including a standard Monte Carlo simulation modelling tool in dynamic spatial environmental modelling languages, would make it much easier for researchers to analyse the errors associated with model outputs, as well as performing sensitivity analysis (Crosetto and Tarantola, 2001).

This chapter focuses on the development of a dynamic spatial environmental modelling language that uses Monte Carlo simulation for error propagation modelling. The set-up of the chapter is as follows. First, the main concepts of error propagation modelling and Monte Carlo simulation are described. Second, a description is given of the extensions which need to be made to the language described in Chapter 3, in order to model error propagation. These extensions involve the entities and functions of the language, and its syntax. Finally, three example models built with a prototype of the language are given to illustrate how it can be applied, followed by a discussion.

4.2 Error propagation modelling in spatial environmental models

Although the emphasis here is on dynamic spatial environmental models, having a time component, error propagation modelling is as important for static spatial environmental models. A static spatial environmental model does not have a time component. It can be described by:

$$A_{1..m} = \mathrm{f}(B_{1..n}, P_{1..j}) \tag{1}$$

with, $B_{1..n}$, the inputs, $A_{1..m}$, the model variables, and a model structure defined by a function or set of functions f, with associated parameters $P_{1..j}$. Note that $B_{1..n}$, $A_{1..m}$ and $P_{1..j}$, are defined in two or three spatial dimensions. An example of a static model is the derivation of infiltration capacity from a soil type map, or the calculation of distance from a vegetation island. As noted in Chapter 3, a dynamic spatial model is described in an analogous way:

$$A_{1..m}(t+1) = \mathrm{f}(A_{1..m}(\mathrm{t}), B_{1..n}(\mathrm{t}), P_{1..k}, t) \tag{2}$$

with the same meaning of the symbols as in equation 1, and t, the time, while the inputs and model variables have a value for each time step, here.

For both types of spatial environmental models, model output error is caused by *input error* and *model error*. The input error is associated with uncertainty in model input variables ($B_{1..n}$(t) in equation 1 and 2) for a given computational model, which are unique for a specific study site or period, such as elevation data, conductivity values or temperature time series data. The model error is associated with the discrepancy between

the computational model, defined by f and $P_{1..j}$ in equations 1 and 2, being an approximation of reality. Model error refers to an incorrect choice of model equations and parameters for the representation of real world processes. Since the analysis of model error is still in development, we focus on the propagation of input error, although the theory and prototype software proposed here can, and will, be used for the analysis of errors in the model parameters, too.

Most studies model error by representing model inputs, parameters, and variables as stochastic (random) fields in two or three dimensions. By doing so, the inputs and parameters of a model ($B_{1..m}$(t) and $P_{1..j}$ in eq. 1 and 2) become stochastic, and also most of the model variables $A_{1..n}$(t) (eq. 1 and 2), because these are derived from the inputs. This kind of model, having stochastic model variables as a result of stochastic inputs and/or parameters, is referred to as a stochastic model. The aim of error propagation modelling, is to derive the probability distributions (or parameters describing these) of the stochastic model variables $A_{1..n}$(t) from the stochastic inputs $B_{1..m}$(t) and parameters, and to store the probability distributions of these model variables of interest, i.e. the model output variables. For complex environmental models, this can only be approximated with Monte Carlo simulation modelling (Heuvelink 1998). Monte Carlo simulation involves two steps (Hammersley and Handscomb 1979, Heuvelink 1998):

Step (1). For each Monte Carlo run s, s=1..K (below, lower case letters represent realizations of random variables given as upper case letters in equation 1 and 2):
(a) generate realizations $p_{1..j}$ of each stochastic model parameter, and realizations $b_{1..m}$ for each stochastic input variable, for each time step t (in the case of a dynamic model),
(b) with these realizations, run f(·) (eq. 1 or 2), and store the realizations of the model output variables $a_{1..n}$, in the case of a dynamic model for all time steps.

Step (2). Compute sample statistics (e.g., mean, variance, skewness) from the K model outcomes, for each model variable 1..n and each time step t, or for a selection of model variables and time steps.

This approach needs a stochastic description of model input variables with their associated errors, and a methodology to draw realizations from these stochastic model inputs, needed in step 1a of the Monte Carlo procedure. A review of the different types of error associated with the input data stored in GIS is given by NCDCDS (1988), Thapa and Bossler (1992), and Lanter and Veregin (1992). For most types of error, stochastic error models and associated numerical techniques for drawing realizations are available. Attribute error for a continuous variable without spatial correlation (assuming a constant attribute value over an area) can be simulated as a random variable with a specified probability density functions using standard algorithms (e.g., Press *et al.* 1986), while standard geostatistical software tools (e.g. Deutsch and Journel 1998, Pebesma and Wesseling 1998) provide algorithms for generating spatially correlated random fields. The same standard tools for geostatistics can be used to generate spatially correlated fields for categorical (classified) attributes, using indicator simulation (e.g., Bierkens and Burrough 1993). Other approaches to model errors in categorical attributes as stochastic variables are provided by Fisher (1991), Goodchild *et al.* (1992), Veregin (1994, 1995). These are mostly based on the classification error matrix (e.g., Lanter and Veregin 1992).

Positional accuracy is mainly important for environmental modelling as a source of class boundary uncertainty, and several stochastic methods have been described to deal with this (Davis and Keller 1997, Kiiveri 1997, Leung and Yan 1998). Another important type of error in environmental modelling is related to uncertainty caused by interpolation of time series data, which seems to be underexposed in research, although it is expected that techniques similar to these used for spatial data apply.

4.3 Extensions to the entities and functions of the language

4.3.1 Entities

As noted in Chapter 3, two entities are used in the language: two dimensional maps, with a spatial discretisation in cells, and three dimensional blocks, with a spatial discretisation in voxels. Each cell on a map contains the same map variables while each cell in a block contains the same block variables. For deterministic dynamic modelling, Chapter 3 proposes a fixed discretisation of time in time steps, with a fixed length of time steps, for all variables. This is represented by a one dimensional array of time steps in which each field contains a value of a variable for a certain map or block. For dealing with Monte Carlo simulation, the same approach is followed, while adding an additional dimension to the array (Figure 4.1). At the end of a model run, each field (t, s) in this array contains an attribute value for time step t and Monte Carlo sample s, for the cell at (x,y), or the voxel V(x,y,v). Note that, in the case of a static model, the two dimensional array with values shown in Figure 4.1 reduces to a one dimensional array, with Monte Carlo samples only.

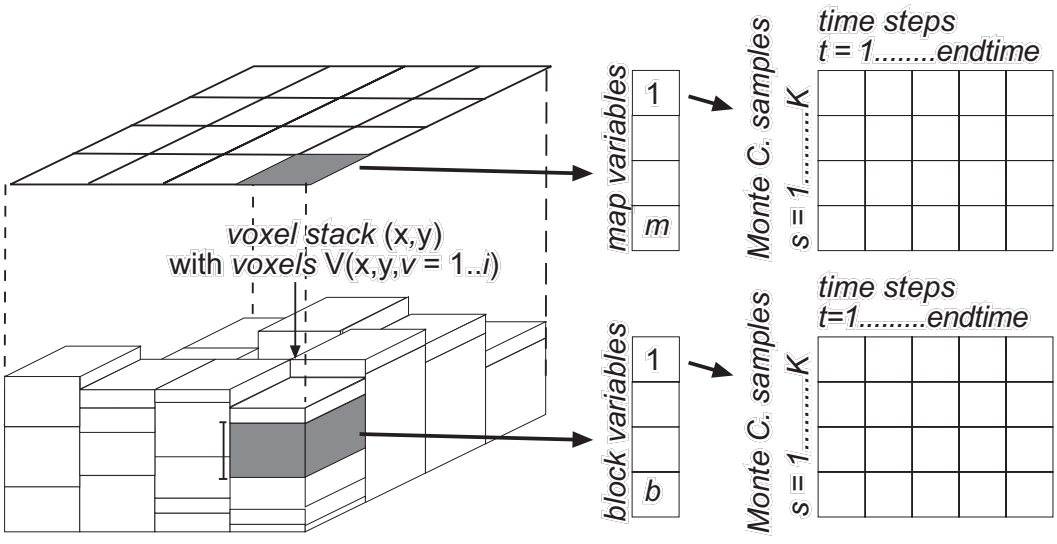


Figure 4.1. Entities of the language. Left, map and block; centre, list of map or block variables; right, matrix with values for each time step and each Monte Carlo sample, stored for each map or block variable, for each cell or voxel.

4.3.2 Functions of the language

Chapter 3 described functions of the language which can be used to represent the function f in equation 1 or 2. For Monte Carlo simulation, stochastic functions and functions calculating descriptive statistics are needed in addition to these functions.

1) stochastic functions
The stochastic functions assign a realization of a random variable to a map or block variable (for step 1a in the Monte Carlo simulation procedure). Different types of model input error are represented by different types of random variables with their associated realizations, as noted in section 4.2.

2) functions calculating descriptive statistics
These functions calculate descriptive statistics of values within a certain group of values in the matrices attached to each cell or voxel (Figure 4.1). Statistics to be calculated include for instance average, quartiles or variance (c.f. Pebesma *et al.* 2001). As shown in Figure 4.2, these statistics can be calculated as an aggregated value 1) over two- or three-dimensional space, for instance average pollutant concentration of a lithological unit, per

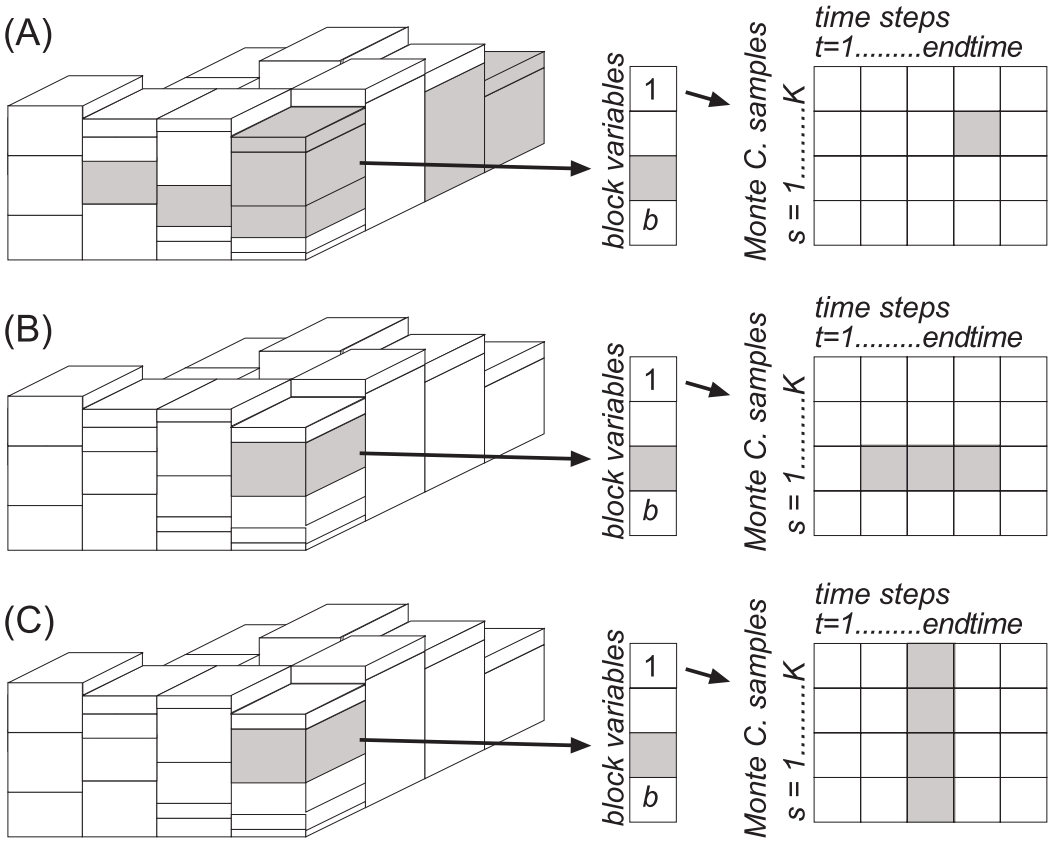


Figure 4.2. Functions calculating descriptive statistics, aggregating over A) space, B) time, C) the stochastic dimension, i.e., Monte Carlo samples.

time step and Monte Carlo sample; 2) over time, for instance the maximum pollutant concentration during a model run, per cell/voxel and per Monte Carlo loop; or 3) over Monte Carlo samples, for instance the variance in pollutant concentration per cell/voxel and time step.

4.4 Syntax

4.4.1 Functions

The syntax of the stochastic functions and the functions calculating descriptive statistics is similar to the syntax of the other functions provided by the language (Chapter 3). The stochastic functions assign a realization of a two or three dimensional random field to a map or block variable, respectively. Their input arguments define the statistical properties of the random field to draw the realizations from. For instance, a realization of a random field with spatial autocorrelation is defined by:

Result=**spatcor**(*Variogram*, $Input_{1..n}$, $Options_{1..n}$)

with, *Result*, the output map or block variable containing a realization of a random field, *Variogram*, a code defining the variogram type and parameters of the random field, $Input_{1..n}$, conditioning data, and $Options_{1..n}$, options (e.g. as applied in Gstat, Pebesma and Wesseling 1998). For other types of random fields, similar notations can be developed.

The functions calculating descriptive statistics aggregate over space, time or the Monte Carlo dimension (Figure 4.2) by calculating statistics such as average, standard deviation, quartiles, for a map or block variable. The statistical functions aggregating over space calculate statistics of cell or voxel values belonging to the same spatial class (Figure 4.2A), for each time step and each Monte Carlo sample. These functions use two inputs: the map or block variable for which the statistical value is calculated, and a map or block variable with classes defining the areas over which statistics need to be calculated. For instance,

```
AveZincLith=spatialaverage(Zinc, Lith)
```

calculates the average zinc concentration (block variable `Zinc`) over each lithological unit (block variable `Lith`), and assigns these average values to the block variable `AveZincLith`. This is done for each time step and each Monte Carlo sample. Using the same syntax, **spatialsd** would calculate the standard deviation of the `Zinc` values per lithological unit. For aggregating over time, functions are provided calculating statistics over periods in time (Figure 4.2B). For instance,

```
AveZincMonth=temporalaverage(Zinc, Month)
```

calculates the average zinc concentration over periods defined by `Month`, which is a block variable containing different identifiers for each month. This monthly average is

calculated for each voxel, and each Monte Carlo sample. The same operation could be done on map variables. Aggregating over the Monte Carlo dimension (Figure 4.2C) is done in a similar way, by calculating statistics over the *K* model outcomes (step 2 in the Monte Carlo simulation procedure), for each time step and each cell or voxel. For instance,

```
ZincQuartile=mcperc(Zinc, 25)
```

calculates the first quartile (defined by the second argument 25) of the zinc concentration Zinc, for each time step and voxel, based on all Monte Carlo samples.

4.4.2 Script structure

The script structure described in Chapter 3 is extended to provide an additional loop, generating the *K* Monte Carlo samples, representing step 1 in the Monte Carlo simulation procedure. Figure 4.3 gives the concepts of the extended script, structured in sections. Each section contains a list of operations which are sequentially executed. The operations in the **preloop** section are executed only once, at the start of the model run. These read data from the data base, generating map or block variables that are the same for each Monte Carlo loop. The **initial** and **dynamic** sections represent the temporal behaviour of the model. The **initial** section defines the initial state of the model at time step 0, reading data from the data base or results of the **preloop** section. The **dynamic** section, representing the temporal change, is a section with functions which are run for each time step. The operations in the dynamic section represent the change in the state of model variables over one time step (eq. 2). For error propagation modelling, the **initial** section and the **dynamic** section (for all timesteps) are run for each Monte Carlo simulation loop. The statistical operations in the **postloop** section aggregate over the spatial, the time, and/or the stochastic (Monte Carlo simulation) domain. Note that a script does not always need to contain all these sections. For instance, in the case of a static model, the dynamic section is not used, as will be shown by the first example model.

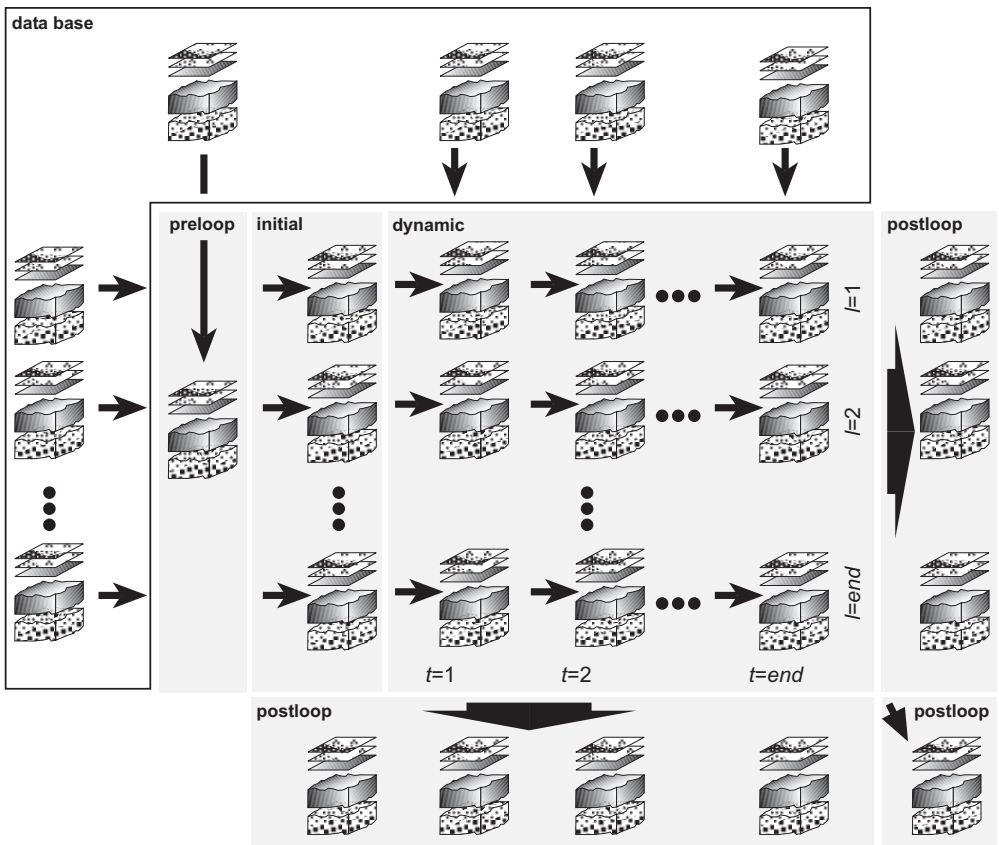
4.5 Example models

4.5.1 2D spatial model

The prototype implementation of the language described in Chapter 3 is extended according to the concepts described in this chapter. Table 4.1 shows a script for a model simulating clonal growth of vegation, made with the prototype implementation of the language. Figure 4.4 gives the most important variables used in the script. A plant species occupies the area given as Boolean TRUE on the map variable `Ini` (Figure 4.4A), read from the database in the preloop section. By clonal growth, the plant spreads over the whole area, resulting in a larger area occupied by the plant after 50 years, which is calculated in the initial section. The first operation in the initial section creates a map containing for each cell the time (yr) needed for the plant to spread one metre. The

value and the error associated with this input parameter depends on the soil type. From a field investigation, the value of the spreading time could be estimated as 0.1 year (standard deviation 0.001) for peat, and 0.5 (standard deviation 0.02) for other soil types. The two lines using the **mapnormal** function, drawing a value from a normal distribution with zero mean and standard deviation one, generate for each Monte Carlo loop a realization of a spreading time map with these statistical properties, independently for peat and for the other soil types, see Table 4.1. These realizations are combined in one

(A)



(B)

```
preloop
  # sequence of spatial functions
  ...

  initial (nrloops = ...)
    # sequence of spatial functions
    ...

    dynamic (nrtimesteps = ...)
      # sequence of spatial functions
      ...

postloop
  # sequence of spatial functions
  ...
```

Figure 4.3. (A) Concepts and (B) modelling script for temporal, two or three dimensional, and error propagation modelling.

```
preloop
  Ini=distr.map            # initial distribution of plant

initial (nrloops=500)
  # spreading time for peat (years)
  PeatYears=0.1+mapnormal()*0.001
  # spreading time for other soil types (years)
  OtherYears=0.5+mapnormal()*0.02
  # number of years needed to move the vegetation front 1 m
  Years=if(Peat then PeatYears else OtherYears)
  # time to colonization (yr)
  ColTime=spread(Ini,Years)
  # colonized after 50 years
  Col=ColTime < 50

postloop
  # probability of colonization after 50 years
  ColProb=mcprob(Col)
```

Table 4.1. Script for modelling clonal growth of vegetation, remarks behind #.

map with an **if..then** function, assigning `PeatYears` to the peat areas having a TRUE on the map variable `Peat` (Figure 4.4B), and `OtherYears` to the other soil types, having a FALSE on the map variable `Peat`. The **spread** function calculates a total spreading time map from the locations occupied with the plant on Ini, multiplying the distance between cells and the 'friction' values on `Years`. The last operation in the initial section assigns a Boolean TRUE to all cells colonized within 50 years. The sequence of operations in the initial is executed 500 times, representing 500 Monte Carlo loops (samples), resulting in 500 realizations of the map variable `Col`. The operation in the postloop section derives from these 500 realizations of `Col`, for each cell the probability that it is occupied by the plant after 50 years, resulting in the map variable `ColProb` (Figure 4.4C).

4.5.2 2D spatial and temporal stochastic model

An example of a temporal model with error propagation analysis is given in Table 4.2. It concerns a hypothetical two dimensional groundwater flow model meant to estimate the depth of the groundwater level below the surface, and the number of days that the groundwater level is in the root zone, which is assumed to be vulnerable for the vegetation. Figure 4.5 gives the most important map variables used in the model. The preloop section reads constant data from the database. These include the digital elevation model of the area, with elevation differences of only a few metres. The transmissivity is the only model input that is not exactly known, but its statistical properties are known. As

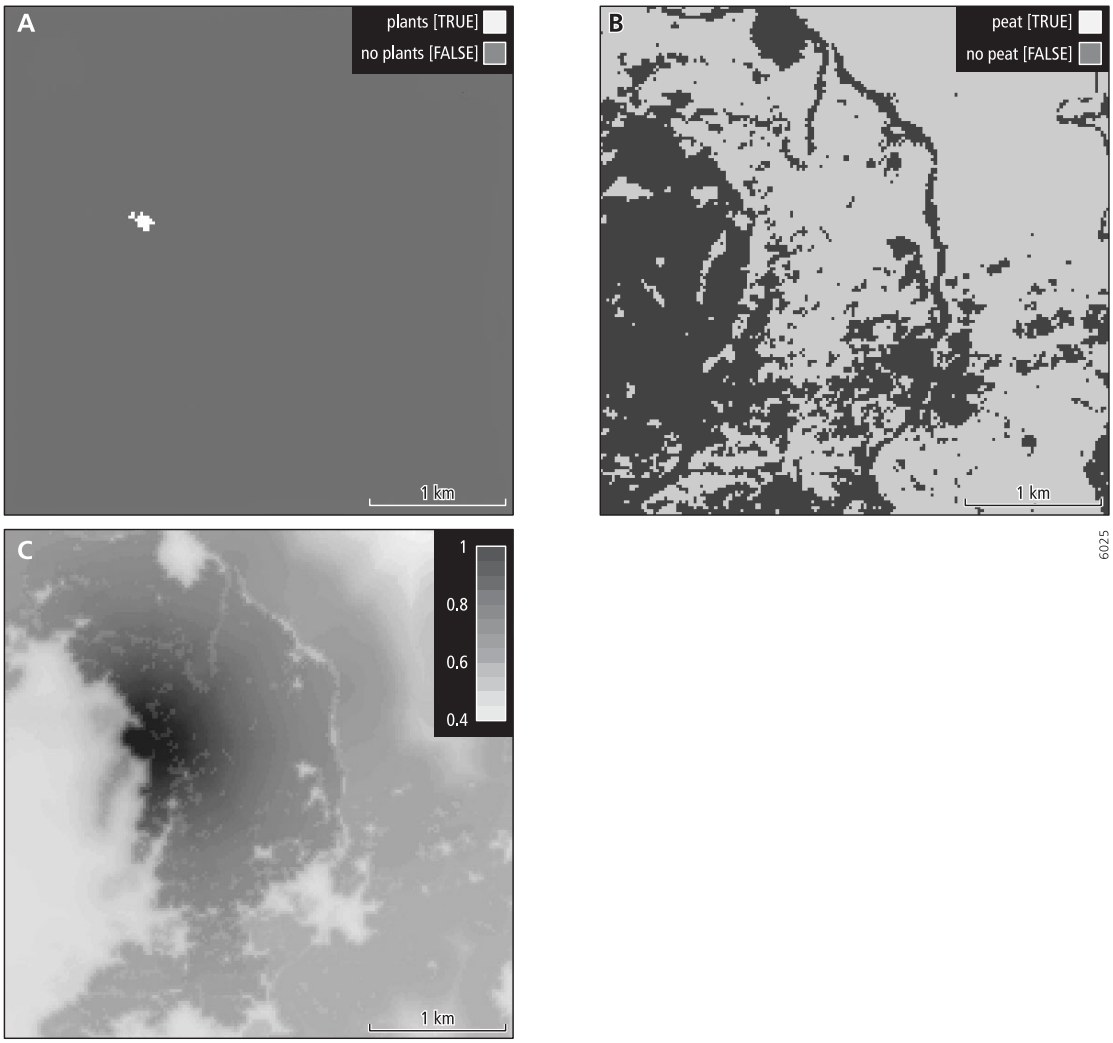


Figure 4.4. Clonal growth model. (A) initial distribution of plant, variable name `Ini` in Table 4.1, Boolean 1 (TRUE) represents growth location, 0 (FALSE) other locations, (B) distribution of peat in the area (`Peat`), Boolean 1 (TRUE) represents peat, o (FALSE) other soil types, (C) probability of colonization after 50 years, `ColProb`.

such, it is the only source of error or uncertainty in the model input, and it is modelled as a two dimensional random field with spatial autocorrelation, described by a variogram. For each Monte Carlo loop, the **spatcor** function in the initial section generates a realization of this random field, using a combined nugget and exponential semivariogram, with a spatial autocorrelation range of 2000 m. The **exp** function calculates the base *e* exponent of this random field, resulting in a lognormal distribution of hydraulic conductivity (Figure 4.10A).

The dynamic section is run for 1600 time steps, with a length of a time step of 0.1 days, explicitly defined with the map variable `T` in the preloop section. The **transient** function at the bottom of the dynamic section simulates transient groundwater flow in an unconfined layer (Crank-Nicolson method, Wang and Anderson 1982). For each time

```
preloop
  Dem=dem.map              # topographical elevation (m)
  FlowCond=cond.map        # flow conditions, flow or fixed
                           # head
  H=hini.map               # head at start of model run (m)
  T=scalar(0.1)            # timestep (days)
  EvapMax=0.001            # maximum evapotranspiration (m/day)
  S=0.2                    # specific yield

initial (nrloops=500)
  # hydraulic conductivity m/day
  K=exp(2.4957+spatcor(0.2Nug(0) + 0.8 Exp(750)))

  dynamic (nrtimesteps=1600)
    # depth of water table (m)
    WTDepth=Dem-H
    # head in root zone
    Wet=WTDepth < 0.45
    # evapotranspiration (m/day)
    Evap=max(EvapMax-0.001*WTDepth,0)
    # recharge (m/day)
    Recharge=max(Rain-Evap,0)
    # head (m)
    Head=transient(Head,Recharge*T,K,FlowCond,S,T)

postloop
  # duration of groundwater in root zone (days)
  Dur=temporaltotal(Wet,1)/T
  # duration of groundwater in root zone, 5% percentile
  Dur5P=mcperc(Dur,5)
  # duration of groundwater in root zone, 95% percentile
  Dur95P=mcperc(Dur,95)
  # significantly above threshold duration of 150 years
  AboveTh=Dur5P > 150
  # significantly below threshold duration of 150 years
  BelowTh=Dur95P < 150
```

Table 4.2. Script for groundwater flow modelling, remarks behind #.

step, it calculates a map with groundwater heads (`Head`, Figure 4.5B), derived from its input maps read from the database in the preloop section, the groundwater recharge (`Recharge`, m), the hydraulic conductivity (`K`, m/day), the flow conditions (`FlowCond`, Figure 4.5A), the specific yield (`S`, dimension less), and the time step (`T`, days). The first function in the dynamic section calculates the depth of the water table below the soil surface, for each time step. The second function creates a map with a Boolean TRUE if the ground water level is closer than 0.45 m below the surface, assumed to be the root zone, otherwise a FALSE. The third function calculates the evapotranspiration as a function of the depth of the water table. The resulting map variable (`Evap`) is used in the fourth operation to calculate the recharge to the

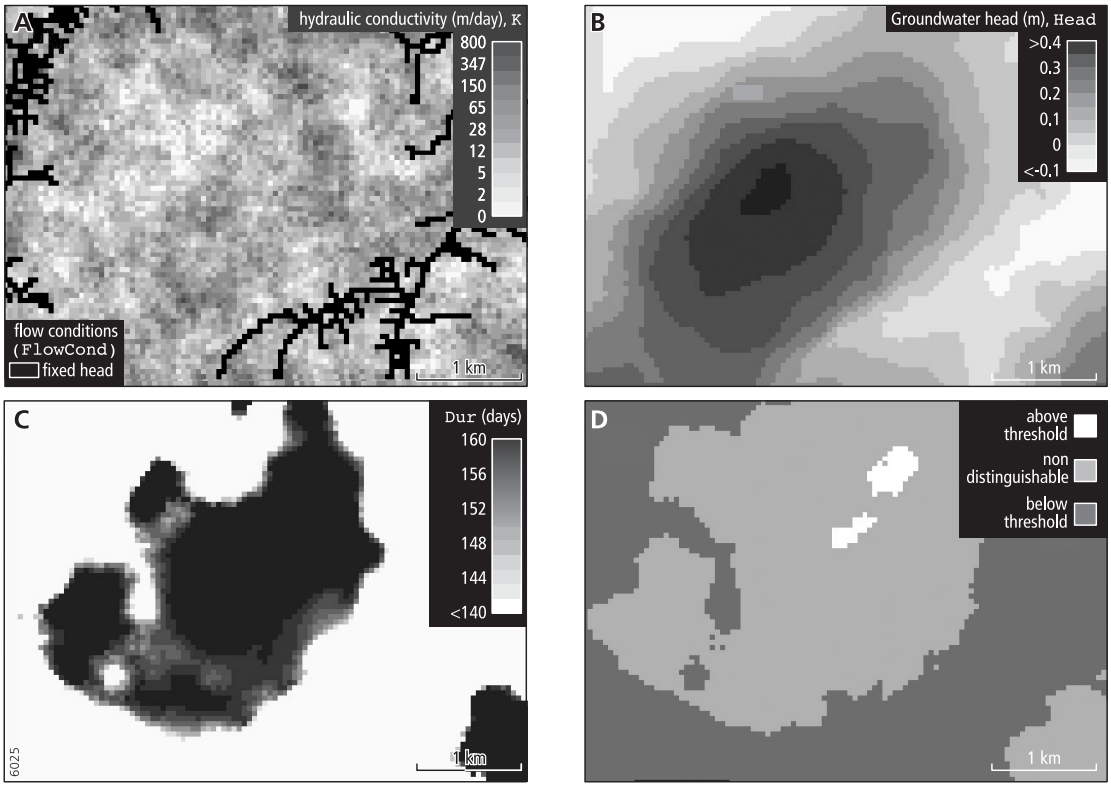


Figure 4.5. Groundwater flow model. (A) realization of the hydraulic conductivity (grey scales, `K` in Table 4.2, m/day), and flow conditions (`FlowCond`, black: river cells with a fixed head, other: flow) (B) realization of the groundwater head (`Head`, metres) at the last time step, (C) realization of the duration of groundwater in root zone (`Dur`, days), (D) areas with duration of groundwater in root zone significantly below or above threshold duration (150 years), and areas for which it is not possible to determine whether the critical level is exceeded or not. Combination of `AboveTh` and `BelowTh` in Table 4.2.

groundwater. The postloop section aggregates over time and over the Monte Carlo dimension. The **temporaltotal** function calculates the total duration (days) that the groundwater level has been in the root zone, for each Monte Carlo sample, resulting in the map variable `Dur` (Figure 4.10C). The next two lines calculate the 5 and 95 percentiles of `Dur`, for each grid cell, using the **mcperc** function. These percentiles define the lower and upper boundary of the 90% confidence interval. For each cell, the probability is 0.90 that the duration of the groundwater in the root zone lies between the `Dur5P` value and the `Dur95P`. By comparing these lower and upper boundaries of the confidence interval with a critical duration of 150 days, maps can be made where the duration of water in the root zone is certainly above (`AboveTh`) or below (`BelowTh`) the critical duration, at a confidence level of 90% (Figure 4.5D).

4.5.3 3D spatial and temporal model

Chapter 3 describes a three dimensional model simulating deposition and erosion in an environment dominated by rivers. It was assumed that all model inputs were exactly known, which is not the case. Table 4.3 shows an extension to the script given in

```
preloop
  Depos=depos.map       # map with deposition rate at
                        # the channel belt (m/timestep)
  In=inflow.map         # inflow location
  Dem=dem.map           # initial elevation model

initial(nrloops=500)

  dynamic (nrtimesteps=20)
    # create a local drain direction map
    Ldd=lddcreate(Dem)
    # centre of the channel belt,
    # path downstream from the inflow point
    Centre=path(Ldd,In)
    # distance to the channel belt centre line (m)
    Dist=spread(Centre,0,1)
    # channel belt with width of 1500 m
    Belt=Dist < 750
    # deposition as a function of distance
    # to channel belt (m/timestep)
    DepCB=if(Belt then Depos else
                      (Depos*exp((750-Dist)/1500)))
    # deposition caused by local scale processes (m/timestep)
    DepLocal=0.05+spatcor(0.01Nug())
    # total deposition (m/timestep)
    Add=DepCB+DepLocal

    Erosion=if(Belt then 10 else 0)
    Deposition=if(Belt then 10+Add else Add)

    Lith=remove(Erosion)
    Lith=add(Deposition,Belt)

    Dem=top()

postloop
  # probability of a channel belt
  BeltProb=mcprob(Lith)
```

Table 4.3. Script for alluvial architecture modelling, remarks behind #.

chapter 3, which includes one additional (hypothetical) input associated with uncertainty: local scale deposition. It is assumed that the total deposition (`Add`, m/timestep, see Table 4.3) for each time step is equal to the sum of deposition determined by large scale processes (`DepCB`, m/timestep) plus deposition determined by small scale processes (`DepLocal`, m/timestep). The large scale deposition is modelled in the same way as in chapter 3, as a function of distance to the channel belt. The small scale deposition is represented by a stochastic variable with a (hypothetical) average value of 0.05, with a spatial pattern defined by a pure nugget variogram with a nugget variance of 0.01, for

which realizations are drawn with the **spatcor** function, also described in Section 4.5.2. This is done independently for each timestep, which means that the pattern of local scale deposition is different for each timestep.

Apart from the lines representing the deposition, the dynamic section of the script in Table 4.3 corresponds to the one given in Chapter 3. The **nrloops** keyword in the initial section sets the number of Monte Carlo loops to 500, which means that the dynamic model, defined in the dynamic section and consisting of 20 time steps, is run 500 times, resulting in 500 realizations of all model variables, for all time steps. As a result of the uncertainty in the deposition, each realization results in a different pattern of channel belts (not shown). The effect of this uncertainty is calculated in the postloop section, by calculating for each voxel the probability of occurrence of channel belt deposits, stored in the block variable `BeltProb`. A two dimensional picture of this block variable is given in Figure 4.6, showing well distinguished zones of low and high probabilities of occurrence of channel belt deposits. The location of the zones with high probability are determined by a fixed pattern of elevation values on the the initial elevation model (`Dem`) and the processes included in the model. Figure 4.10 in chapter 7 gives three dimensional pictures generated with a similar, but more complicated, model.

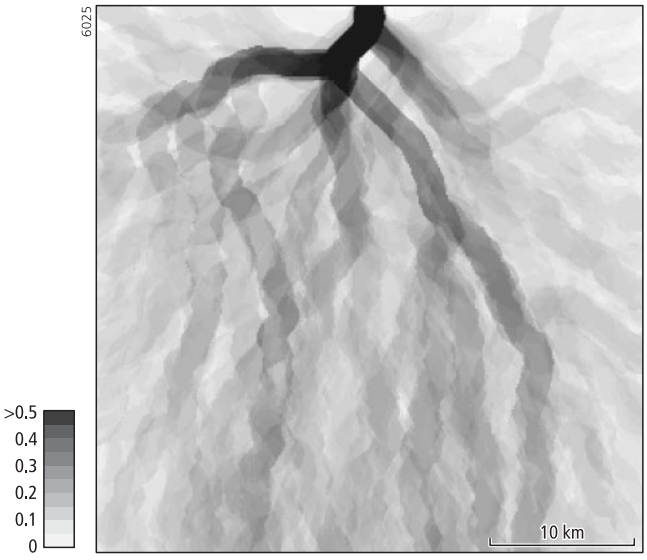


Figure 4.6. Alluvial architecture model, probability of occurrence of a channel belt, `BeltProb` in Table 4.3. Plan view of the block variable `BeltProb`, showing the highest probability found below each x,y location.

4.7 Discussion and conclusions

Although the example models demonstrate that modelling error propagation is possible using the language, many improvements need to be made for a wider, and better, application. Compared to the language described in Chapter 3, the amount of data that

have to be managed by this language is even larger, because an additional Monte Carlo dimension has been added. Also, the sections in the script are executed in a fixed order, from top to bottom. This results in large amounts of (sometimes) unused data stored on hard disk, while calculation times may become excessively high. Optimisation algorithms need to be developed that manage data storage, restricting data storage to 1) variables for which the model builder explicitly defines in the script that they need to be stored, and 2) those variables, time steps or Monte Carlo samples for which the results need to be stored since they are needed in a later phase script execution. In addition, algorithms need to be developed that find the optimal order of calculation of dynamic models in a Monte Carlo simulation scheme, e.g., whether the time steps should be nested in the loops of the Monte Carlo simulation or vice versa. The optimisation schemes regarding data storage and calculation order need to parse the whole script before it is executed.

The multidimensionality of the data dealt with has also a major impact on the visualisation routines needed to explore model inputs and outputs. Exploratory data analysis techniques and software for temporal, multidimensional data associated with environmental models hardly exist. It might be needed to apply visualisation techniques that go beyond these known in cartography or standard statistics, such as glyphs, which are graphical objects whose elements (e.g., position, size, shape, colour, orientation) are bound to data (Foley and Ribarsky, 1994).

Acknowledgements
We wish to thank Edzer Pebesma, Peter Burrough, Willem van Deursen, Marc Bierkens, and Cees Wesseling for providing many inputs to the research described here.

4.8 References

Bierkens, M.F.P. & P.A. Burrough (1993), The indicator approach to categorical soil data. I. Theory. Journal of Soil Science 44, pp. 361-368.

Crosetto, M. & S. Tarantola (2001), Uncertainty and sensitivity analysis: tools for GIS-based model implementation. International Journal of Geographical Information Systems 15, pp. 415-437.

Davis, T.J. & C.P. Keller (1997), Modelling uncertainty in natural resource analysis using fuzzy sets and Monte Carlo simulation: slope stability prediction. International Journal of Geographical Information Systems 11, pp. 409-434.

Deutsch, C.V. & A.G. Journel (1998), GSLIB Geostatistical Software Library and Users's guide, second edition. New York: Oxford University Press.

Fisher, P.F. (1991), Modelling soil map-unit inclusions by Monte Carlo simulation. International Journal of Geographical Information Systems 5, pp. 193-208.

Foley, J. & B. Ribarsky (1994), Next-generation Data Visualisation Tools. In: Scientific Visualisation. Advances and Challenges (eds L. Rosenblum et al.), pp. 103-126. Academic Press, London.

Goodchild, M.F., S. Guoqing & Y. Shiren (1992), Development and test of an error model for categorical data. International Journal of Geographical Information Systems, 6, pp. 87-104.

Hammersley, J.M. & D.C. Handscomb (1979), Monte Carlo Methods. London: Chapman & Hall.

Heuvelink, G.B.M. (1998), Error Propagation in Environmental Modelling with GIS. London: Taylor & Francis.
Kiiveri, H.T. (1997), Assessing, representing and transmitting positional uncertainty in maps. International Journal of Geographical Information Systems 11, pp. 33-52.
Lanter, D.P. & H. Veregin (1992), A research paradigm for propagating error in layer-based GIS. Photogrammetric Engineering & Remote Sensing 58, pp. 825-833.
Leung, Y. & J. Yan (1998), A locational error model for spatial features. International Journal of Geographical Information Systems, 12 pp. 607-620.
NCDCDS (1988), The American Cartographer 15, pp. 11-140.
Pebesma, E. & C.G. Wesseling (1998), Gstat: a program for geostatistical modelling, prediction and simulation. Computers & Geosciences 24, pp. 17-31.
Pebesma, E. J., D. Karssenberg & K. de Jong (2001), The stochastic dimension in a dynamic GIS. In: Proceedings of Compstat 2000, 14th Conference of the International Association for Statistical Computing, 21-25 August, Utrecht, the Netherlands.
Press, W.H., S.A. Teukolsky, W.T. Vetterling & B.P. Flannery (1986), Numerical recipes in Fortran 77, the art of scientific computing, first edition, Cambridge: Cambridge University Press.
Thapa, K. & J. Bossler (1992), Accuracy of spatial data used in Geographic Information Systems. Photogrammetric Engineering & Remote Sensing 58, pp. 835-841.
Veregin, H. (1994), Integration of simulation modeling and error propagation for the buffer operation in GIS. Photogrammetric Engineering & Remote Sensing 60, pp. 427-435.
Veregin, H. (1995), Developing and testing of an error propagation model for GIS overlay operations. International Journal of Geographical Information Systems 9, pp. 595-619.
Wang, H.F. & M.P. Anderson (1982), Introduction to groundwater modeling. Finite difference and finite element methods. New York, USA: Academic Press.

PART II:

UPSCALING

5. UPSCALING OF SATURATED CONDUCTIVITY UNDER STEADY STATE HORTONIAN RUNOFF

Derek Karssenberg

Abstract: A stochastic scaling model is described for calculating effective saturated conductivity of the topsoil under steady state conditions of rainfall, infiltration and Hortonian runoff. The model predicts an increase in effective saturated conductivity of a model domain with 1) an increase in rain intensity, 2) increasing mean and decreasing variance, skewness and spatial scale of saturated conductivity within the model domain, 3) increasing size of the model domain and, 4) decreasing bifurcation of the drainage pattern in the model domain. Using saturated conductivity derived from ringinfiltrometer measurements, the scaling model predicts values of effective saturated conductivity for larger plots (1 m^2) which are comparable with effective saturated conductivity values derived from rainfall simulations on these plots. It is shown that the effective saturated conductivity values predicted by the scaling model for a hillslope (7500 m^2) can be used in a simulation of runoff from that hillslope with reasonable results. The differences between simulated and measured cumulative discharge from the hillslope are mainly caused by periods of rainfall-runoff conditions deviating from the steady state conditions assumed by the scaling model.

5.1 Introduction

The actual infiltration under Hortonian overland flow depends on the flow process in the soil with a boundary condition at the surface set by the water available on the surface. The flow process in the soil is determined by hydraulic properties of the soil having considerable spatial variation, at all scales (*Blöschl and Sivapalan*, 1995). The water available at the surface consists of water from rain which is spatially and temporally variable as a result of variable rain and interception, and run-on, with spatial variation determined by surface flow patterns, and temporal variation during a storm event.

Because of this spatial variation in the processes of infiltration, the conceptualization, and/or the associated inputs, outputs, state variables and parameters in a model of infiltration need to change with the size of the spatial and temporal domain for which the process is represented by the model, also called the scale of the model (*Blöschl and Sivapalan*, 1995). A scaling rule for this change would allow comparison of results from different measurement techniques of infiltration at different sizes of the domain, ranging from 0.002-0.05 m^2 when ring infiltrometers are used (e.g., *Sharma et al.*, 1980; *Sullivan et al.*, 1996; *Smettem*, 1987), up to 0.5-10 m^2 for plot experiments under artificial or natural rainfall (e.g., *Sharma et al.*, 1980; *Sullivan et al.*, 1996; *Smettem*, 1987; *Yu et al.*, 1997). In addition, such a scaling rule is needed to parameterize model conceptualizations of infiltration applied in catchment models, using the above mentioned small scale field measurements of infiltration. This is because the size of the model domain of the infiltration model in a catchment model is in most cases larger than the size of the domain represented by field measurements. Typically, the size of the model domain represented

by the infiltration equations in a catchment model is either a hillslope or field unit (e.g., *Knisel and Williams*, 1995; *Smith*, 1995), or a grid cell on a hillslope with run-on from upstream cells (e.g., *de Roo*, 1996).

A scaling rule for infiltration needs to quantify the variability of small scale processes occurring within the larger scale model domain (*Blöschl and Sivapalan*, 1995; *Blöschl*, 1996, *Bierkens et al.*, 2000). The first approach, applied in perturbation techniques and parameterisation, assumes that the conceptualization of the process of infiltration needs to change from a smaller to a larger model domain. This will also involve a change in model inputs, outputs, state variables and/or parameters. The second approach is to use the same conceptualization of infiltration at all scales, with a scale rule involving only a change in the parameter value(s) with scale. This approach uses either distribution functions of parameter values or effective parameter values at the larger scale. An effective parameter is the single value assigned to all locations within a model (sub-) domain such that the model based on that value yields the same average output as the model based on the actually occurring heterogeneous parameter field (*Blöschl and Sivapalan*, 1995).

The concept of effective parameters will be applied in this study mainly because of its simplicity. Other research indicates that effective parameters can be applied to steady state situations with constant model inputs, outputs and state variables. Both with ring infiltrometer measurements and plot experiments with artificial rainfall, a steady state of infiltration is reached since inputs of water at the top boundary can be kept constant throughout the experiment, resulting in a steady state situation, with constant matrix and macro-pore flow, referred to as the saturated conductivity for the steady state situation under consideration. Because of this, it can be expected that the effective parameter concept provides a scale rule between ring infiltrometer and plot experiments. For natural rainstorms and catchments, steady state conditions do not hold, and the effective parameter concept can only be applied under the assumption that effective parameters derived for steady state conditions are also valid under transient conditions.

Existing methods dealing with effective parameters (e.g., *Russo*, 1992; *Hendrayanto et al.*, 2000) mostly ignore the interaction between surface runoff and infiltration or do not deal with spatially variable runoff (e.g., *Grant et al.*, 1991; *Yu et al.*, 1997; *Corradini et al.*, in press), although this interaction is known to be important (*Smith and Hebbert*, 1979; *Woolhiser et al.*, 1996; *Merz and Plate*, 1997; *Corradini et al.*, 1998; *Binley and Beven*, 1989). *Merz et al.* (in press), comparing different approaches for modeling plot scale infiltration, find best results when spatial variation of runoff within the plot is taken into account. The method described here derives an effective saturated conductivity for steady state infiltration under Hortionian overland flow, representative for a model domain as a function of 1) the spatial probability distribution of saturated conductivity within the model domain, 2) the spatial pattern of surface flow within the model domain, under steady state conditions, 3) the size and shape of the model domain, and 4) the rain intensity.

The setup of the paper is as follows. First, the model for deriving values of effective saturated conductivity is developed. Next, the fundamental behavior of the model is illustrated with simple artificial catchments and a comparison with trends found in other studies. After that, in a case study, using data from the Ouvèze river basin (S. France), the validity of the model in field studies will be evaluated. First, ring infiltrometers are used in the model to estimate values of effective saturated conductivity for model domains

with the size of plot experiments. These effective values are compared with saturated conductivity values derived from measurements at that plot scale, using artificial rainfall plot experiments. Second, different approaches are evaluated to apply the effective saturated conductivity values in a transient simulation of runoff from a hillslope, in order to test the hypothesis that effective values derived with a steady state model can be applied in transient simulations. The paper ends with conclusions and recommendations.

5.2 Scaling model for steady state infiltration

5.2.1 Introduction

For a small area $u \in \Re^2$ of the modeling domain, with |u| typically 0.04 m^2, it can be assumed that water available for infiltration, rain (P_u, mm/h) and run-on (Q_u, mm/h), is evenly spread over the modeling domain and the actual infiltration I_u (mm/h) under steady state conditions can be conceptualized as:

$$I_u = \min(P_u + Q_u, I_{max}) \quad (1)$$

with I_{max}, the maximum infiltration rate (i.e., infiltration capacity) of the area u. The function min(a,b) is defined as assigning the minimum value of a and b. Under steady state conditions, the soil is saturated for the case when the term $P_u + Q_u$ is greater than I_{max}, and I_{max} equals the saturated conductivity of the top soil of the area u. So, under steady state conditions, equation (1) can be rewritten as:

$$I_u = \min(P_u + Q_u, K_u) \quad (2)$$

with K_u the saturated conductivity of the area u. In many cases, the model inputs are not exactly known, and need to be represented as random variables, represented here by upper case letters. At larger spatial scales, i.e. a plot or a hillslope with an area l, it is assumed that the same conceptualization of infiltration is valid, although the small scale saturated conductivity K_u needs to be replaced by the effective saturated conductivity K_e (mm/h) at the larger scale l:

$$I_l = \min(P_l + Q_l, K_e) \quad (3)$$

with I_l, the steady state actual infiltration (mm/h) at the larger scale l, and P_l, the rain (mm/h), Q_l the run-on (mm/h) at the larger scale. The aim is to develop a model deriving K_e from the following information: 1) parameters describing the spatial probability distribution of K_u within the area l, 2) parameters describing the surface flow pattern under steady state conditions within the area l, 3) the size |l| and shape of the area l, and 4) the rain intensity.

5.2.2 Stochastic representation of saturated conductivity and flow pattern

A landscape with spatial variation in saturated conductivity is represented by the random field (i.e. random function) $\{K(\mathbf{s}): \in \Re^2\}$, where $\mathbf{s}$ is a spatial coordinate in the two dimensional domain. In the model, flow over the surface and infiltration are assumed to occur over square units u($\mathbf{s}$). Since the area |u| (m^2) of each unit is chosen to be very small, typically 0.04 m^2, it is assumed that rain on a unit and inflow from upstream to the unit is evenly distributed over the whole unit, as noted above. Under this assumption, the saturated conductivity at the support of a unit $K_u(\mathbf{s})$ is the average of $K(\boldsymbol{s})$ over the unit u:

$$K_u(\mathbf{s}) = \frac{\int_u K(\mathbf{s})\mathrm{d}\mathbf{s}}{|u|} \tag{4}$$

At the scale of the modeling domain with an area |u|, the plausible assumption can be made that $K_u(\mathbf{s})$ has a lognormal distribution (*Russo and Bresler*, 1981; *Ragab and Cooper*, 1993; *Buttle and House*, 1997; *Mallants et al.*, 1997):

$$K_u(\mathbf{s}) = e^{Z(\mathbf{s})}. \tag{5}$$

The random field $\{Z(\mathbf{s}): \in \Re^2\}$ with a normal distribution has the following properties:

$$m_{Z(\mathbf{s})} = \mathrm{E}\{Z(\mathbf{s})\}, \tag{6}$$

$$\gamma(\mathbf{h}) = \frac{1}{2}\mathrm{E}\left\{\left(Z(\mathbf{s}) - Z(\mathbf{s}+\mathbf{h})\right)^2\right\}, \tag{7}$$

The quantity $m_{Z(\mathbf{s})}$ is the expectation (mean) of $Z(\mathbf{s})$. The quantity $\gamma(\mathbf{h})$ which is a function only of the separation vector $\mathbf{h}$, is called the semivariogram defining the spatial structure of $Z(\mathbf{s})$. In the absence of other information, a spherical model is assumed for the semivariogram:

$$\gamma(\mathbf{h}) = \begin{cases} b\left(1 - \frac{3}{2}\frac{|\mathbf{h}|}{a} + \frac{1}{2}\frac{|\mathbf{h}|^3}{a^3}\right) & \text{for } 0 \le |\mathbf{h}| \le a \\ b & \text{for } |\mathbf{h}| > a \end{cases} \tag{8}$$

with; a, range of the semivariogram defining the spatial scale of variation (m); b, maximum value of the covariance. The expectation (mean) of $K_u(\mathbf{s})$ is (*Aitchison and Brown*, 1957):

$$\mathrm{E}\{K_u(\mathbf{s})\} = m_{K_u(\mathbf{s})} = e^{m_{Z(\mathbf{s})} + \frac{1}{2}b}, \tag{9}$$

Increasing b results in a probability distribution of $K_u(\mathbf{s})$ with a higher skewness.

In this paper, surface flow is represented by flow over a rectangular grid of units u($\mathbf{s}_i$), with $\mathbf{s}_1,\ldots,\mathbf{s}_n$, the locations of the center of the units at a regular grid, with a grid spacing corresponding to the length and width of the units. At this grid, realizations of $K_u(\mathbf{s}_i)$ are generated by sequential Gaussian simulation with Latin hypercube sampling (*Pebesma and Heuvelink*, 1999). The model assumes that the exact flow pattern between units is not known, although general characteristics will be known, such as the large scale change in topography and the presence or absence of micro relief such as furrows. The model generates realizations of the flow pattern derived from a digital elevation model defined as a random field with characteristics representing the large scale topography as a deterministic surface and micro relief added as a random component with specified spatial correlation characteristics. For each realization, the model derives the runoff pattern by assigning to each unit a flow direction to the neighboring unit resulting in the steepest downstream slope, removing small, local depressions (8-point pour algorithm, *Burrough and McDonnell*, 1998).

5.2.3 Process model

For determining effective saturated conductivity, a steady-state process model is used with the stochastic inputs described above. Any unit draining to a unit u($\mathbf{s}_i$) is called a neighboring upstream unit u($\mathbf{s}_{i,n}$), n=1..m (Figure 5.1). Assuming a known rain intensity p (mm/h), equation 2 becomes:

$$I(\mathbf{s}_i) = \min(p + Q_{in}(\mathbf{s}_i), K_u(\mathbf{s}_i)), \tag{10}$$

$$Q_{in}(\mathbf{s}_i) = \sum_{n=1..m} Q_{out}(\mathbf{s}_{i,n}), \tag{11}$$

with (see also Figure 5.1) $I(\mathbf{s}_i)$, the actual infiltration of the unit u($\mathbf{s}_i$) (mm/h); $Q_{out}(\mathbf{s}_{i,n})$, outflow received by unit u($\mathbf{s}_i$) from a neighboring unit (mm/h); $Q_{in}(\mathbf{s}_i)$, total inflow (i.e., runon) from neighboring units 1..m to unit u($\mathbf{s}_i$) (mm/h); and $K_u(\mathbf{s}_i)$ saturated conductivity of unit u($\mathbf{s}_i$) (mm/h). All fluxes are stochastic variables, apart from the rain intensity, denoted with a lower case letter p. Outflow ($Q_{out}(\mathbf{s}_i)$, mm/h) from a unit is:

$$Q_{out}(\mathbf{s}_i) = Q_{in}(\mathbf{s}_i) + p - I(\mathbf{s}_i). \tag{12}$$

For each unit u($\mathbf{s}_i$) its catchment is defined as the unit itself and all its upstream units. The K_e value (eq. 3) for the catchment of a unit is:

$$K_e(\mathbf{s}_i) = \frac{p \cdot N(\mathbf{s}_i) - Q_{out}(\mathbf{s}_i)}{N(\mathbf{s}_i)}, \tag{13}$$

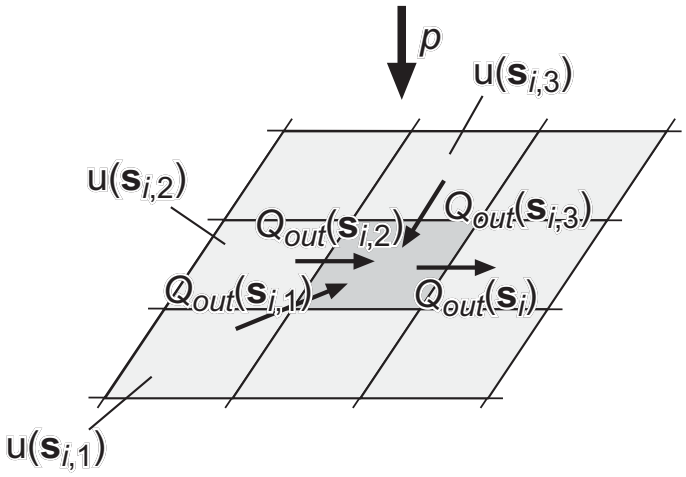


Figure 5.1. Rain p and surface flow between square units. Each unit drains to one neighboring unit. Example unit u($\mathbf{s}_i$) in the center receives runoff $Q_{out}(\mathbf{s}_{i,1..3})$ from neighboring units u($\mathbf{s}_{i,1..3}$).

with $N(\mathbf{s}_i)$, number of units in catchment of u($\mathbf{s}_i$); $K_e(\mathbf{s}_i)$, effective saturated conductivity of the catchment of u($\mathbf{s}_i$) (mm/h).

For the outflow point u($\mathbf{s}_{out}$) of a catchment, $K_e(\mathbf{s}_{out})$ is derived from: (1) the model given by equations (10-13), (2) the distribution of the input random field $K_u(\mathbf{s}_i)$ given by equations (4-9) and the random surface flow pattern; (3) the rain intensity, constant in space and time. The Monte Carlo simulation approach solves this in two steps (*Hammersley and Handscomb*, 1979; *Heuvelink*, 1998). Step 1. Repeat M times: a) generate a realization of the input random field $K_u(\mathbf{s}_i)$ and the flow pattern between the units u($\mathbf{s}_i$), b) with this realization of $K_u(\mathbf{s}_i)$ and the flow pattern run the model given by equations (10-13) for a given rainfall intensity p and store the model outcome (realization) of $K_e(\mathbf{s}_{out})$. Step 2. Compute sample statistics (e.g., median, percentiles) of $K_e(\mathbf{s}_{out})$ from the M model realizations of $K_e(\mathbf{s}_{out})$. Both steps can be repeated for different distributions for $K_u(\mathbf{s}_i)$, different flow patterns and different rain intensities. All calculations described were done with a value of M higher than needed to retrieve sample statistics (Step 2) at the precision required.

5.3 The fundamental behavior of the scaling model

5.3.1 Evaluating one realization

To illustrate how the model behaves, it is applied to a domain consisting of a simple linear catchment of 50 units following each other. For the sake of clarity, one realization of $K_u(\mathbf{s}_i)$ is taken to illustrate the effect of increased rain intensity on infiltration (Figure 5.2A) and effective saturated conductivity (Figure 5.2B). At lower rainfall intensities not all units infiltrate at the rate of the saturated conductivity since the total input to these units, rainfall plus inflow from upstream cells, is less than the saturated conductivity. At these lower rainfall intensities, the effective infiltration rate of the catchment is lower than the mean saturated conductivity of the catchment. With increasing rain intensity, the

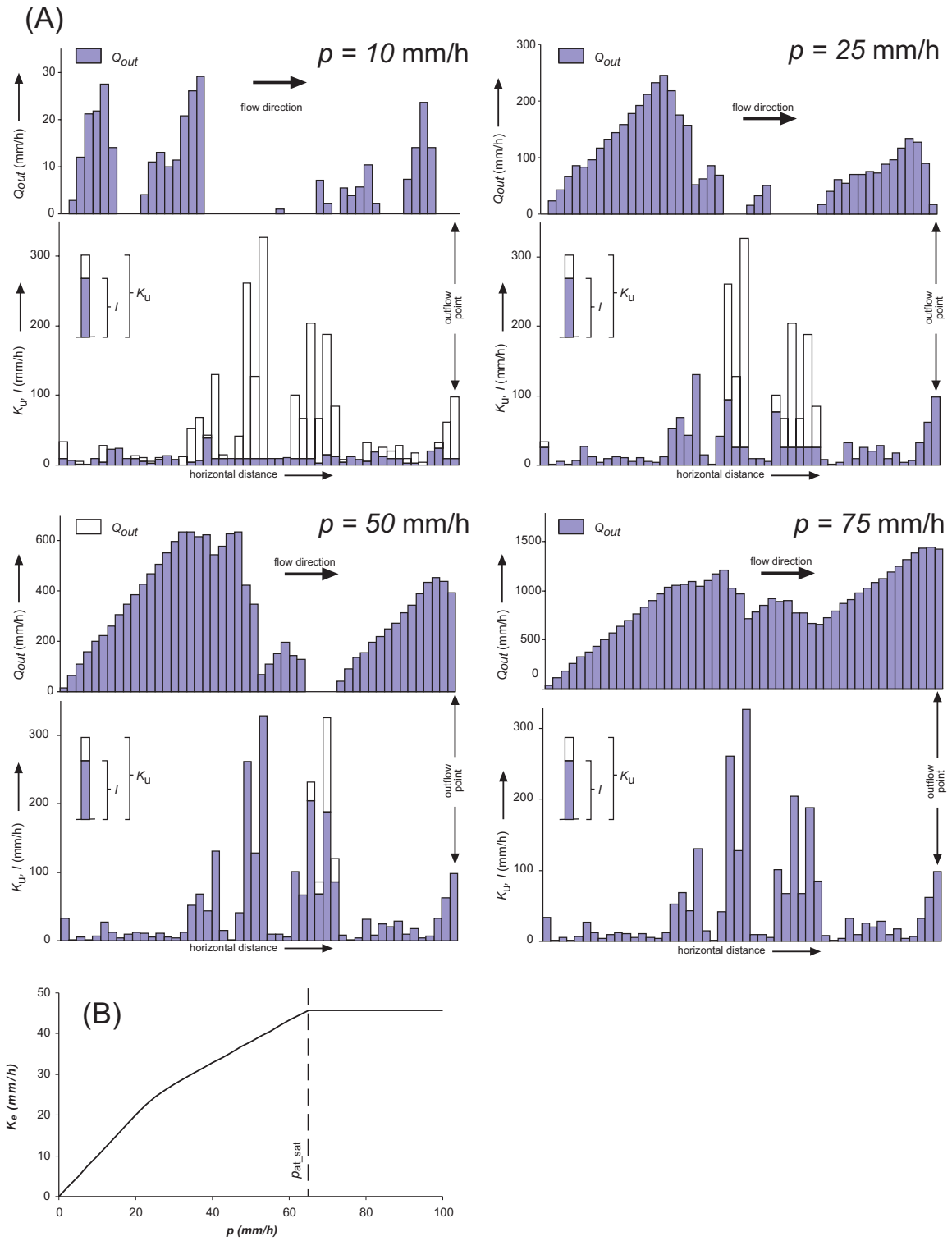


Figure 5.2. Model input and outputs for one realization of $K_u(\mathbf{s}_i)$ for a linear catchment of 50 units. (A) Fluxes (mm/h per unit area) per unit from upstream to downstream at different rain intensities p, K_u, saturated conductivity; I, actual infiltration; Q_{out}, outflow. (B) Effective saturated conductivity ($K_e(\mathbf{s}_{out})$, mm/h) at different rain intensities (p, mm/h) for the catchment of the outflow point.

infiltration rate increases since more water is available for infiltration. This increase stops at rain intensity p_{at_sat} where all units infiltrate at the saturated conductivity (Figure 5.2B). Above this rain intensity, the effective saturated conductivity corresponds to the mean of the saturated conductivity of the catchment. For this example realization, the value of p_{at_sat} and the increase of k_e with p depends on the distribution and spatial pattern of the saturated conductivity. The Monte Carlo simulation applied in the model permits the calculation of the full probability distribution of the effective saturated conductivity, as described below.

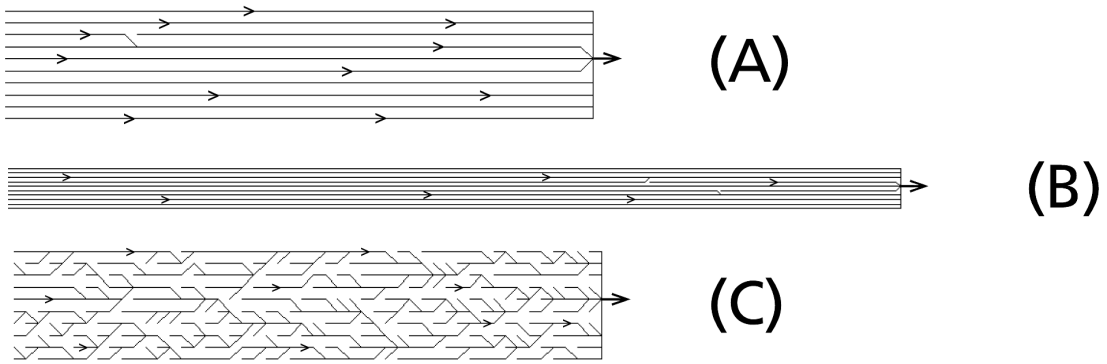


Figure 5.3. Example realizations of flow patterns, plan view. (A) Catchment of 10 x 50 units, used for *base*, *less skewed* and *larger range* scenarios; (B) catchment of 10 x 500 units, used for *larger catchment* scenario; (C) network catchment of 10 x 50 units, used for *network* scenario.

5.3.2 Sensitivity analysis

Although effective saturated conductivity is a function of several parameters, the focus in this paper is on its value as a continuous function of rain intensity (p, mm/h). This relationship is shown for different values for other input parameters using an artificial data set. All distances are given here in 'units' length, where one unit length is the length $\sqrt{|\mathrm{u}|}$ of a model unit. Different scenarios of input parameters are compared with a *base* scenario which calculates the effective saturated conductivity of a model domain (catchment) consisting of 50 x 10 units of 'unit' length. This catchment has a constant slope in the longest direction plus a very small amount of spatially uncorrelated random noise, resulting in parallel flow paths in one preferential direction, with locally some convergent flow (Figure 5.3A). The probability distribution of the saturated conductivity within this model domain is defined by the expected value of saturated conductivity ($\mathrm{E}\{K_u(\mathbf{s}_i)\}$ = 50 mm/h) and its skewness defined by the b parameter (b = 2.0, eq. 9), having values that could also occur in the field. The spatial distribution of saturated conductivity is defined by the range parameter a of the semivariogram (eq. 8), which is for the *base* scenario 3 units length, meaning that the scale of variation in the 'base' scenario is much smaller than the length of the catchment for which the effective saturated conductivity is calculated. Although this standard catchment is scale free, we could think of it as a hillslope with parallel sheet flow and spatial variation in saturated

Table 5.1. Model parameters. For each scenario, the parameter $m_{Z(\mathbf{s})}$ is derived with equation 7. Bold letters are parameter values different from the base scenario.

Scenario	*a* nr. unit lengths	*b*	E{ $K_u(\mathbf{s}_i)$} mm/h	Nr. units -	Flow pattern -
base	3	2.0	50	10 x 50	sheet
less skewed	3	0.5	50	10 x 50	sheet
larger range	15	2.0	50	10 x 50	sheet
larger catchment	3	2.0	50	10 x 500	sheet
network	3	2.0	50	10 x 50	network

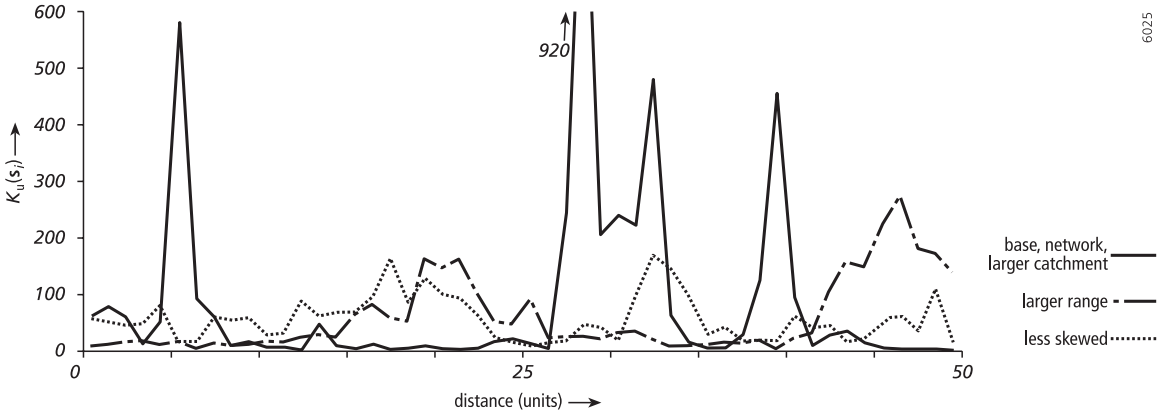


Figure 5.4. Example realizations of $K_u(\mathbf{s}_i)$ for the scenarios in Table 5.1, shown as transects of 50 units.

conductivity over a distance of several meters. This *base* scenario is compared with 4 other scenarios, each of which has one parameter changed compared to the *base* scenario. Inputs for all scenarios are given in Table 5.1 and Figure 5.3. The results are based on a Monte Carlo with M=999 loops. Figure 5.4 gives example realizations of $K_u(\mathbf{s}_i)$ shown as a transect over 50 units.

The results in Figure 5.5 show the relationship between effective saturated conductivity $K_e(\mathbf{s}_{out})$ and rainfall intensity for the different scenarios. Since $K_e(\mathbf{s}_{out})$ is a random variable, both the median and the 45th and 55th percentiles are given. A large difference between these percentiles implies that $K_e(\mathbf{s}_{out})$ shows much variation, which means that model domains (catchments) having the same properties with respect to the model input parameters have large differences in $K_e(\mathbf{s}_{out})$. The following factors determine $K_e(\mathbf{s}_{out})$, other simulations (not shown) with different changes in input parameters give comparable results:

1. Rain intensity. Increasing the rainfall intensity results in an increase in effective saturated conductivity for all scenarios. At rain intensity values below 20 mm/h, which is much lower than the expectation of saturated conductivity in the catchment (E{$K_u(\mathbf{s}_i)$} = 50 mm/h), the effective saturated conductivity is very close to, but lower than, the rain intensity. At higher rain intensities, the effective saturated conductivity is lower than the rain intensity and approaches E{$K_u(\mathbf{s}_i)$}. An increase in effective saturated conductivity

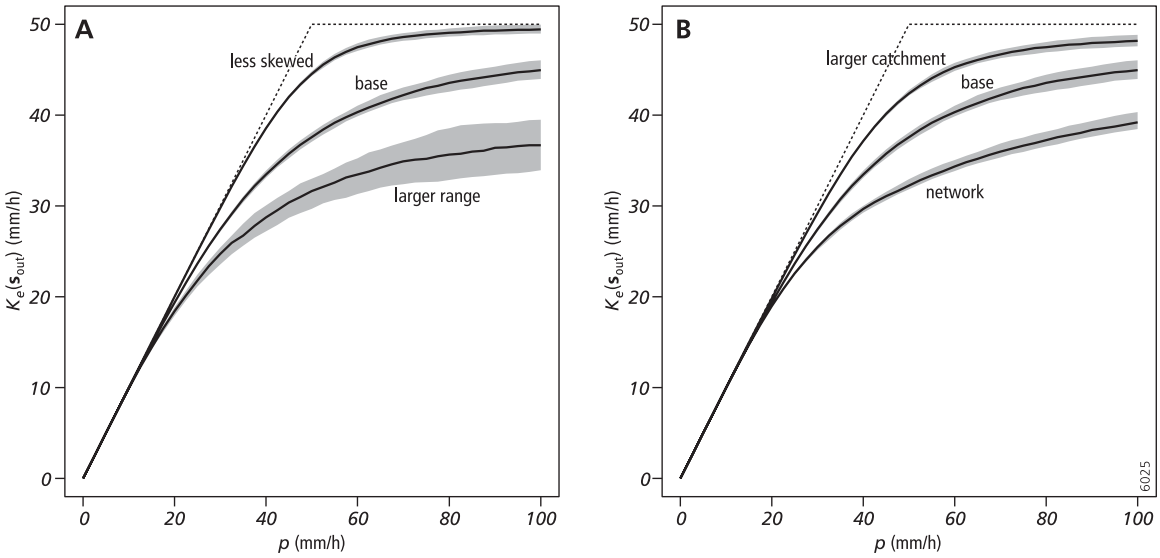


Figure 5.5. Effective saturated conductivity of a model domain ($K_e(\mathbf{s}_{out})$, mm/h) against rain intensity (p, mm/h). Each figure compares the *base* scenario with two other scenarios: (A) *less skewed*, *base*, and *larger range* scenarios, (B) *larger catchment*, *base*, and *network* scenarios. Solid line, median; gray area, values between 45th and 55th percentile. Dashed line represents effective saturated conductivity without spatial variation in $K_u(\mathbf{s}_i)$ within the model domain.

with rain intensity up to a maximum of $E\{K_u(\mathbf{s}_i)\}$ was also found in the model of *Binley and Beven*, 1989 and in measured and modeled saturated conductivity of *Yu et al.* (1997) and *Merz et al*. (in press). Fields studies by *Burt* (1998), *Harms and Chanavsky* (2000), and *Joel et al.* (in press) also suggest an increase in effective saturated conductivity with rain intensity.

2. The probability distribution of saturated conductivity $K_u(\mathbf{s}_i)$. The effective value of the saturated conductivity approaches $E\{K_u(\mathbf{s}_i)\}$ if p goes to infinity. Decreasing the value of b (eq. 7), implying a decrease in variance and skewness of $K_u(\mathbf{s}_i)$, results in an increase of effective saturated conductivity at all rain intensities, as shown by the *less skewed* scenario (Figure 5.4B, 5.5A). A further decrease in b, would cause the line of effective saturated conductivity to approach and finally (at $b = 0$) overlap the dashed line in Figure 5.5, representing the effective saturated conductivity without any variation of $K_u(\mathbf{s}_i)$ within the model domain. The model of *Corradini et al*. (1998) also predicted a decrease in effective saturated conductivity with decreasing variation and skewness in $K_u(\mathbf{s}_i)$. Several field studies showed effective values of saturated conductivities to be significantly lower than the mean of $K_u(\mathbf{s}_i)$ within a natural or artificial catchment (e.g., *Williams and Bonnel*, 1988), which corresponds with the model simulations presented here. An additional effect of decreasing the variance and skewness of $K_u(\mathbf{s}_i)$ is a decrease in variation in $K_e(\mathbf{s}_{out})$, as shown by the decreased difference between the 45th and 55th percentiles.

3. Spatial distribution of saturated conductivity $K_u(\mathbf{s}_i)$. An increase in the scale of spatial variation results in a decrease in effective saturated conductivity as shown by the *larger range* scenario with a larger scale of spatial variation in saturated conductivity (Figure 5.5A). In addition, the difference between the 45th and 55th percentiles increases. A

comparable decrease in K_e was also found by the modeling study of *Corradini et al.* (1998) and to my best knowledge not corroborated by field studies.
4. Size of the model domain. Increasing the size of the model domain results in an increase in effective saturated conductivity. This is shown by the *larger catchment* scenario (Figure 5.5B), having a catchment of 500 x 10 units (Figure 5.3B) instead of 50 x 10 units. The same trend was suggested by the modeling study of *Blöschl et al.* (1995) and found in field studies of *Harms and Chanasyk* (2000), *van de Giesen et al*. (2000), and *Joel et al.* (in press).
5. Runoff pattern. The *network* scenario is different only from the *base* scenario in its runoff pattern. It represents a runoff pattern with more surface roughness, simulated by a higher amount of spatially uncorrelated noise added to the elevation model. This results in a runoff network with more convergent flow than the *base* scenario, as shown in the realization of runoff patterns in Figure 5.3C. Figure 5.5B shows that the *network* scenario results in a lower effective saturated conductivity than the *base* scenario. This is explained by regarding the results for the *network* scenario as an average of multiple small catchments. Small catchments have a lower effective steady state infiltration rate than large catchments, as was shown by the *larger catchment* scenario.

5.4 Case study: scaling from local to plot scale

5.4.1 Introduction

The scaling model provides a rule to scale saturated conductivity from a scale where it can be conceptualized as a local process without spatial interaction using K_u for saturated conductivity (eq. 2), to the scale where it needs to be conceptualized as a spatial process, with K_e for saturated conductivity (eq. 3). To test the validity of the scaling model scaling from K_u to K_e, it needs inputs of saturated conductivity derived from field measurements at the local, point, scale, while its outputs at the larger scale need to be compared with estimates of saturated conductivity derived from field measurements measuring infiltration as a spatial process at that scale. In this case study, estimates of the probability distribution of saturated conductivity derived from ring infiltrometer experiments, denoted as K_u^*, will be used as input K_u in the scaling model, in order to estimate the probability distribution of effective saturated conductivity $K_{e,p}$ for plots of 1 m^2. This scaled value of $K_{e,p}$ is compared with the saturated conductivity, denoted as $K_{e,p}^*$, derived from artificial rainfall experiments, in order to validate the scaling model.

The study area comprises the 'La Folie' catchment in the Ouvèze river basin, S. France, which is a 0.5 km^2 cultivated catchment with hillslope angles of 0.25 m/m on average, near the village of Entrechaux. It contains loamy sands, sandy loams, loams and silt loam soils (USDA classification), all thicker than 2 m over bedrock. The catchment has a sub-Mediterranean climate, with a mean annual rainfall of 800 mm.

5.4.2 Field measurements

Eighteen infiltration measurements on silt loam soils in vineyards were made with double-ring infiltrometers in May and June, 1997. The inner ring was 20 cm in diameter and the outer one was 40 cm in diameter. Rings were driven 3-5 cm into the ground and smeared with cement at the side, to minimize leakage. Water was applied simultaneously to both inner and outer rings, keeping a constant head of 3 cm in the ring. Infiltration in the inner ring was recorded by readings from a Mariotte bottle every 5 seconds. The experiments were stopped after 0.03 m cumulative infiltration in the inner ring. Saturated conductivity (k_u^*, mm/h) was derived by fitting the a (mm/h) and s (mm/h$^{0.5}$) parameters in the equation (*Philip*, 1969, 1987), assuming $a = 2/3 \cdot k_e^*$ (Green-Ampt soil, *Philip*, 1969, 1987):

$$i_c = s\sqrt{t} + at \tag{14}$$

with i_c, the cumulative infiltration (mm) in the inner ring at time t (h). Results are shown in Table 5.2.

Effective steady state infiltration rates for small plots at 173 randomly chosen locations on cultivated land in the study area were obtained by rainfall simulation experiments, done in the period May-July, 1994-1997. Portable rainfall simulators were used on rectangular plots bounded at three borders with metal strips preventing runoff over the borders. At the fourth, downstream border, runoff was collected in a gutter, sealed to the plot surface with cement. Runoff in the gutter was measured under a rainfall intensity which was kept constant during an experiment, at time intervals of 0.5-2 minutes, until a constant rate of runoff was measured, assumed to represent a steady state situation of infiltration in the plot. The actual unit kinetic energy (UKE), representative for the potential of the raindrops to detach soil material, was calculated from drop size and velocity distributions with an optical spectro-pluviometer (*Salles and Poesen*, 1999). For measuring infiltration for different plot sizes at different rain intensities, multiple rainfall simulators were used (Table 5.3), using a different constant rain intensity for each experiment. For each experiment, $k_{e,p}^*$ was calculated as the infiltration rate on the plot during steady state conditions of rainfall, infiltration and runoff, at the last part of the experiment.

The distribution of K_u^* derived from the ring infiltrometer measurements can be regarded as log-normal. A Kolmogorov-Smirnov test rejected the hypothesis of a normal distribution, while a log-normal distribution was not rejected ($\alpha = 0.05$), and a cumulative

Table 5.2. Summary statistics for saturated conductivity (K_u^*, mm/h) measured by ring infiltrometers. Model parameters derived from these statistics are $m_{K_u^*}$ = 1002 mm/h (mean of K_u^*) and b = 7.73 (variance of ln(K_u^*+1)). s.d., standard deviation, c.v., coefficient of variation.

	mean	median	s.d.	c.v. (%)	n
K_u^*	1002	280	1502	150	18
ln(K_u^*+1)	5.081	5.629	2.780	55	18

Table 5.3. Properties of rainfall simulators. Type of rainfall generating mechanism s: spray nozzles, d: drop-formers (Battany and Grismer, 2000); UKE, unit kinetic energy or kinetic energy applied to 1 m^2 of the soil surface by 1 mm of artificial rain based on 5 observations in each plot.

Simulator	Type	Rain intensity (mm/h)	UKE ($J \cdot m^{-2} \cdot mm^{-1}$)	Plot width, length (m, m)
1	s	25-90	13	0.8, 1.0
2	s	58-79	13	0.8-1.2, 2.1-2.6
3	s	85-105	17	0.6, 0.6
4	d	20-30	7	0.4, 0.8
5	s	67-152	7	0.4, 0.8
6	d	80-423	23	0.14, 0.14

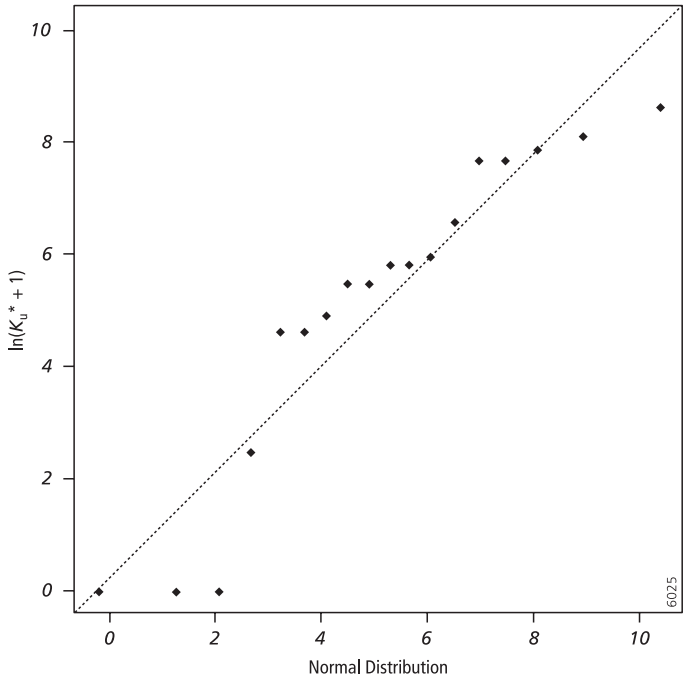


Figure 5.6. Quantile-quantile plot of natural logarithm of saturated conductivity ($\ln(K_u^*+1)$) of ring infiltrometer measurements. Horizontal axis, normal distribution with mean 5.081 and standard deviation 2.780, corresponding with summary statistics of $\ln(K_u^*+1)$. $n = 173$.

probability plot gives a rather good correspondence with a log normal distribution (Figure 5.6). Log normal distributions are almost always found for effective saturated conductivity. The mean of the values is large compared to results found in other studies (e.g., *Russo and Bresler*, 1981; *Sullivan et al*., 1996; *Sharma et al.*, 1980; *Smettem*, 1987; *Loague and Gander*, 1990), which can be explained by the occurrence of macro pores. *Lauren et al*. (1988) and *Williams and Bonnel* (1988) found comparably large values, when macro pores were present in the soil. Figure 5.7 shows that effective saturated conductivity $K_{e,p}^*$ derived with the rainfall simulators increases with rain intensity. *Burt*

(1998), *Yu et al.* (1997), *Merz et al.* (in press), *Joel et al.* (in press) report a comparable relationship between effective saturated conductivity and rain intensity on plots.

The general trends found in the data set correspond with these simulated by the model presented in this study. An increase in $K^*_{e,p}$ (Figure 5.7) with rain intensity was also found in the sensitivity analysis with the scaling model. This suggests that processes described by the model are the key processes determining the measured trends in effective saturated conductivity. Although the rainfall simulator experiments showed a correlation with unit kinetic energy ($r^2 = 0.46$), it is not expected that removal of surface crusts at high rainfall intensities, as suggested by *Burt* (1989), plays an important additional role in increasing effective saturated conductivity with rain intensity. This is because 1) the occurrence or removal of surface crusts was not observed during the rainfall simulations, 2) the correlation with unit kinetic energy, a measure for the potential of the rainfall to remove crusts, was weak. This suggests that removal of crusts is subordinate to the process of runon-runoff interacting with a within plot variation of saturated conductivity.

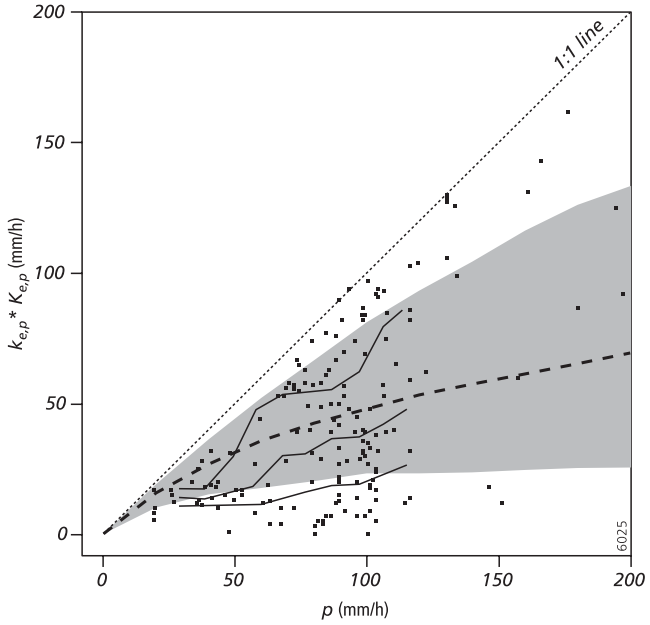


Figure 5.7. Effective saturated conductivity against rain intensity for plots. Dots: $k^*_{e,p}$ (mm/h) derived from rainfall plot experiments against rain intensity applied with the rainfall simulator (p, mm/h), n = 173. Solid lines: first quartile, median and third quartile of $k^*_{e,p}$, moving average with width of moving window of 50 mm/h. Dashed line: median of modeled effective saturated conductivity ($K_{e,p}$, mm/h) against rain intensity (p, mm/h), gray area represents values between modeled first and third quartile.

5.4.3 Upscaling

The model was used for upscaling the ring infiltrometer measurements to effective saturated conductivity ($K_{e,p}$) of rainfall simulation plots. This was done with a model domain catchment area of 1 x 1 m, representing the area of rainfall simulation plots for

which effective values are calculated, and a model unit size |u| of 0.2 x 0.2 m, representing the approximate area of ring infiltrometer measurements. The parameters defining the probability distribution of the random field $K_u(\mathbf{s}_i)$ (equation 8-9) were assigned the measured values of the probability distribution K_u^* of the ring infiltrometer measurements; $m_{K_u^*}$ = 1002 mm/h and b = 7.73 (Table 5.2). It was assumed that variation in $K_u(\mathbf{s}_i)$ can be represented by the spherical semivariogram model in equation 8. Since the spatial scale of variation in $K_u(\mathbf{s}_i)$, represented by the range parameter a in equation 8, was not measured, this parameter was calibrated as described below. Since detailed elevation data were only available for the hillslope used for the transient simulation, these data were used to derive runoff patterns for simulating the rainfall plot experiments. For each Monte Carlo loop, the local drain direction pattern was derived from the elevation model of a 1 x 1 m plot at a random location on this hillslope.

The aim is to find the range parameter a resulting in the smallest deviation between the probability distribution $K_{e,p}^*$ given by the measurements $k_{e,p}^*$ (Figure 5.7) and the modeled probability distribution $K_{e,p}$, at all rain intensities. For each rainfall experiment, the model estimates $K_{e,p}$, at a rain intensity p corresponding to the intensity applied during the rainfall experiment. This was done with 200 Monte Carlo loops for each rainfall experiment, which resulted in a simulated distribution of saturated conductivity at different p, which could be plotted as dots just like the $k_{e,p}^*$ values in Figure 5.7, although it would contain 173 x 200 dots. The simulated and measured values were compared by making decile-decile plots of the measured distribution $K_{e,p}^*$ and simulated distribution $K_{e,p}$, in three intervals i of p: 1) 0-50, 2) 50-100 and 3) 100-200 mm/h. The range parameter a in equation 6 was calibrated using these decile-decile values by minimizing the sum of squares SS:

$$SS = \sum_{i=1}^{3} \left\{ \sum_{j=1}^{9} \left(D_{i,j} - D_{i,j}^* \right)^2 \right\} \qquad (15)$$

with i, the interval number of p; j, the decile number; $D_{i,j}$ the observed, measured, decile (mm/h); $D_{i,j}^*$ the modeled decile (mm/h). Calibration resulted in a range parameter a of 1.2 m (Figure 5.8 and 5.7), with a value for SS (eq. 15) of 3248 mm^2/h^2.

Figure 5.7 shows that, after calibration of the range parameter, the modeled distribution $K_{e,p}$ for the rainfall plots fits well with the measured distribution $K_{e,p}^*$, although the model somewhat overestimates percentile values at rain intensities in interval 1 and 2, while it underestimates percentiles in interval 3 (Figure 5.8). Although the calibrated range of 1.2 m cannot be verified with the field measurements presented here, it is a plausible value. Experimental semivariograms of saturated conductivity at the soil surface given in *Buttle and House* (1997), *Lauren et al.* (1988), and *Loague and Gander* (1990) suggest that 50-100% of spatial variation occurs over distances smaller than 5-10 m. The correspondence between the modeled and measured distribution of effective saturated conductivity for plots suggests that processes described by the model are indeed the key processes determining the measured trends in effective saturated conductivity.

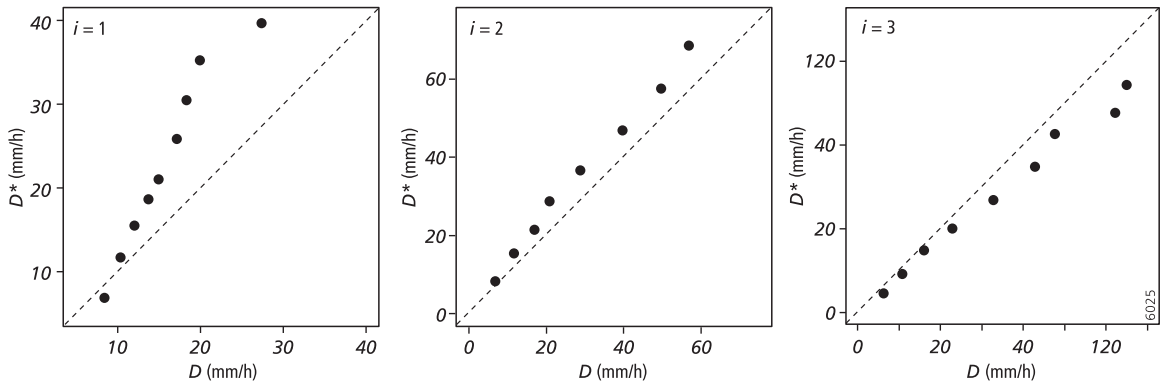


Figure 5.8. Modeled (D^*) against observed (D) deciles for three intervals (i=1..3) of rainfall intensity of rainfall simulation experiments.

5.5 Case study: scaling from local to hillslope scale.

5.5.1 Field measurements and upscaling

For rainfall runoff models representing a catchment by a set of, internally homogeneous, connected hillslope segments, effective saturated conductivity values are needed for each hillslope segment. To test the applicability of the scaling model in such rainfall runoff models, it is used to scale the estimates of saturated conductivity derived from the ring infiltrometer measurements to effective saturated conductivity $K_{e,h}$ of one hillslope in the same study area. The effective values for the hillslope will be applied in a rainfall runoff model for that hillslope using natural rainstorms as input, in order to test the applicability of the scaled values, theoretically only valid under steady state conditions, in transient conditions. Different approaches will be tested to deal with the transient conditions.

Event based runoff modeling will be done for one hillslope, which is an approximately rectangular 7500 m^2 sandy loam vineyard in the study area, with a mean slope of 0.042 m/m. The direction of tillage on the hillslope is approximately parallel to the elevation contour lines. Field data to derive model input parameters and variables were collected in 1997 and 1998 (Table 5.4, c.f., *Van Dijck*, 2000). Errors in water depth and velocity head measurements by continuous pressure recorders in a flume were estimated resulting in an estimated measured cumulative discharge d_m from the hillslope (m^3) and minimum ($d_{m,min}$) and maximum ($d_{m,max}$) values of estimated cumulative discharge, apart from hydrographs. Data were available for 16 events in 1996 and 1997. Since both the ring infiltrometer and the rainfall simulation measurements did not reveal any significant difference in K_e^* between fields in the area, it is assumed that the input parameters for $K_u(\mathbf{s}_i)$ derived from these measurements ($m_{K_u^*}$ = 1002 mm/h and b = 7.73, a = 1.2 m) can also be applied to the hillslope. The effective saturated conductivity ($K_{e,h}$) under the assumption of steady state conditions was calculated using the same model unit size of 0.2 x 0.2 m. The drainage pattern is derived from the elevation model of the hillslope plus random noise (Monte Carlo simulation, step 1b). For simulating the effect of different agricultural practices, three scenarios of random noise added to the original

Table 5.4. Source of parameter values for runoff modeling on the hillslope. All data collected on the hillslope itself, or adjoining similar, hillslopes, except where indicated.

Model component	Parameter	Unit	Source	N
Catchment representation	elevation, drain directions per grid location	m	interpolation from elevation contours	-
Rain	rain intensity	m/h	tipping bucket, Casella, 0.077 mm/tip	1 location, 300 m from the hillslope
Interception	vegetation cover	m^2/m^2	field measurement of canopy area and vertical photographs of canopy	Several in winter and summer
	maximum content of interception store	m	through fall measurements after rainstorms	20
Infiltration	net capillary drive	mm	rainfall simulations	173
	volumetric rock content	m^3/m^3	3500 cm^3 soil samples	31
	saturated moisture content	m^3/m^3	100 cm^3 soil samples	114
	initial moisture content	m^3/m^3	local source*	-
Surface storage	maximum surface storage	mm	based on field measurements and Kamphorst et al. 2000	
Surface runoff	discharge from hillslope	m^3/h	flume with continuous pressure recorders	-
	Manning's n		Chow (1959)	outflow point from the hillslope

* Centre d'Information Régional Agrométéorologique et Economique, Carpentras

digital elevation model for each Monte Carlo loop were used (Figure 5.9). These scenarios are 1) sheet flow, with random noise consisting of a realization from a random field given by a spherical semivariogram with range = 1000 m and variance 0.00005 m, 2) micro scale variation, with random noise consisting of spatially correlated variation given by a spherical semivariogram with range = 0.5 m and variance 0.05 m, 3) flow with wheel tracks, consisting of parallel, straight wheel tracks in a direction D (degrees), with a constant interval between the wheel track centers of 1.25 m, a wheel track width of 0.5 m, and a depth of 0.01 m. The direction D is a random variable with an average direction corresponding to the average direction of the elevation contour lines on the hillslope and a variance of 1. For each Monte Carlo loop, a different wheel track pattern was obtained by generating the wheel track pattern using the above given constants and a realization of D.

Figure 5.10 gives the results, which were derived with a number of Monte Carlo loops M=200. Since each realization in the Monte Carlo procedure results in an almost similar relation between p and $k_{e,h}$, the spreading in the distribution of $K_{e,h}$ is small, as shown in Figure 5.10. The figures shows that the difference in $K_{e,h}$ between the three different scenarios of runoff patterns is small compared to the change in $K_{e,h}$ with rain intensity.

5.5.2 Application in a transient simulation

5.5.2.1 *Rainfall runoff model and scenarios*. The event based rainfall runoff model uses grid cells of 10 x 10 m and a time step of 5 seconds. All input parameters were kept constant in space over the hillslope. For each time step, the net rainfall is the rainfall minus interception at that time step, calculated according to Merriam (1973). For each cell, and each time step, actual infiltration is the minimum value of the saturated conductivity (k) at the time step and the amount of net rainfall plus run-on from upstream cells at the time step. The value of k is derived for each time step with the scaling model with the spatial probability distribution of the saturated conductivity as input, as will be

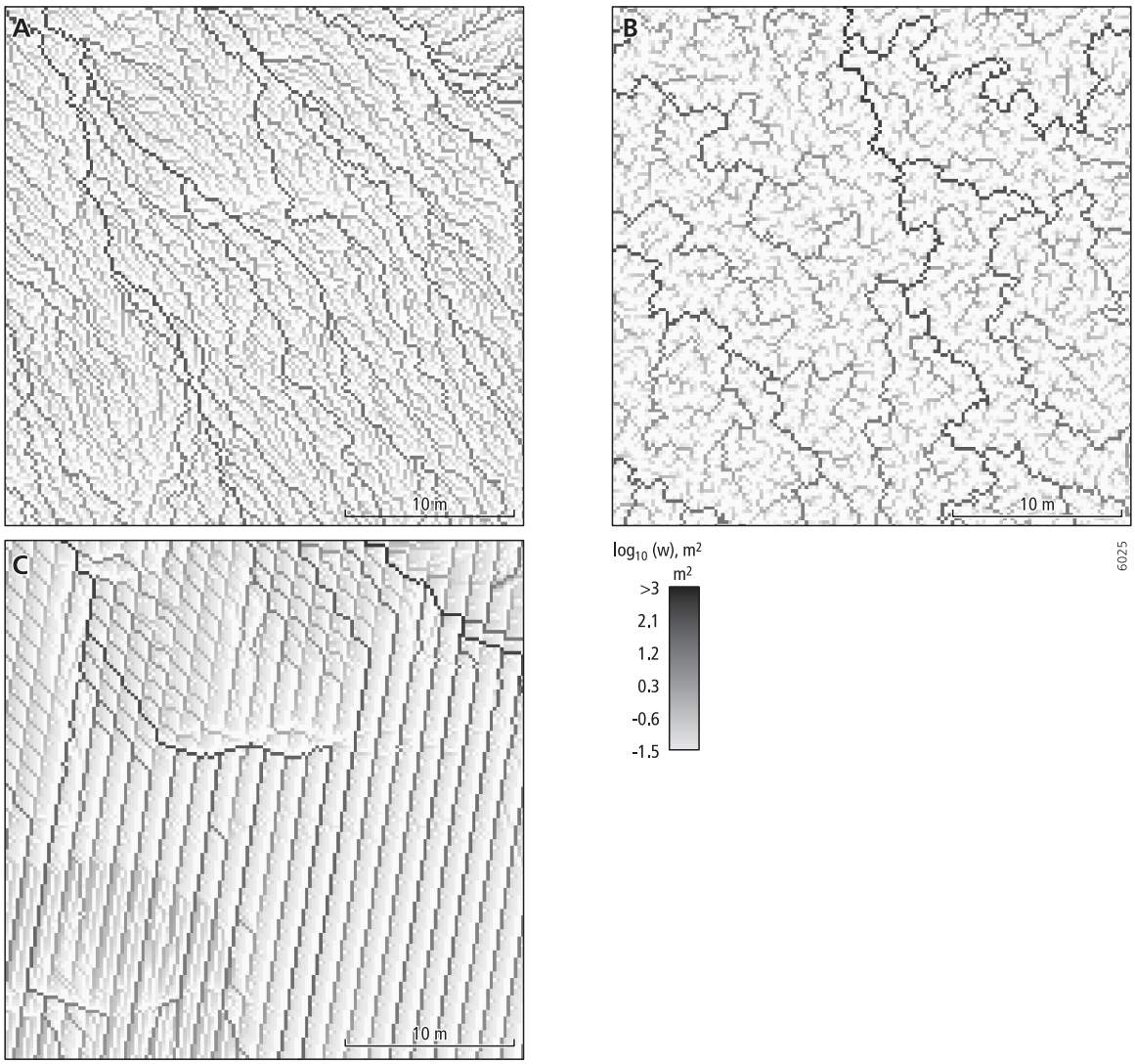


Figure 5.9. Example realizations of drainage patterns for a zoomed area of the field. Catchment area (w) of each unit u($\mathbf{s}_i$). (A) sheet flow, (B) micro scale variation, (C) wheel tracks.

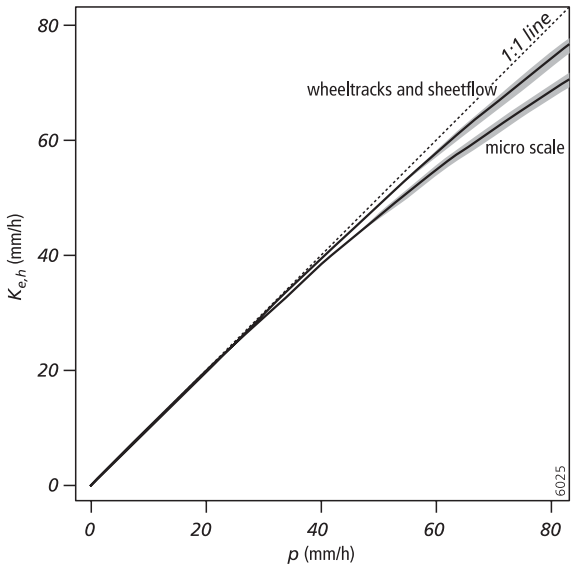


Figure 5.10. Modeled effective saturated conductivity ($K_{e,h}$, mm/h) against rain intensity for the hillslope for three scenarios of runoff patterns. Solid line: median; gray area represents values between modeled first and third quartile. Lines for wheel tracks and sheet flow approximately correspond.

described below. It is assumed that values of maximum infiltration higher than the saturated conductivity at the start of a rainstorm as a result of the suction force in an unsaturated soil can be ignored. Surface water that is not kept in a constant potential surface storage is routed as sheet flow with the kinematic wave using the Manning equation (*Li et al.*, 1975; *Chow et al.*, 1988). The model was implemented in the PCRaster spatio-temporal modeling language (*Wesseling et al.*, 1996).

The relationship between p and the median of the K_e^* distribution derived from the upscaling procedure with wheel tracks (Figure 5.10) was used for deriving the value of k. In the *PIntEvent* scenario, the k value is kept constant throughout the event, corresponding to the K_e^* value read from the p-median(K_e^*) curve (Figure 5.10) at a p value corresponding to the mean of the rain intensity during measured runoff. In the *PIntVar* scenario, the k value varies with t, the real time (seconds) with respect to a reference time for an event. The $k(t)$ value is read from the p-median(K_e^*) curve using the average rain intensity between $t_d(t)$ and t, with

$$t_d(t) = \begin{cases} t - d(t) & \text{for } t - d(t) > t_s(t) \\ t_s & \text{for } t - d(t) \le t_s(t) \end{cases} \qquad (16)$$

with $d(t)$, the average travel time of water from the edge of the hillslope to the outflow point at t, and $t_s(t)$, the maximum of all $t_d(t)$ values in the past. In addition, three scenarios with the same k value for all events were run. These are the *Ring* scenario, with k = 1002 mm/h corresponding to the mean value of the saturated conductivity K_e^* derived from the ring infiltrometer measurements, the *Plot* scenario, with k = 46 mm/h, the mean saturated conductivity $K_{e,p}^*$ derived from the rainfall simulations, and the *InvMod* scenario, with k = 1.7 mm/h derived from inverse modeling with the rainfall runoff model, based on a

minimization of the mean of the squared differences between measured cumulative discharge (d_m, m^3) and simulated cumulative discharge (d_s, m^3) of the events.

5.5.2.2 *Results and discussion*. Table 5.5 gives summary statistics of the 16 events. Table 5.6 gives for each scenario the mean error (*ME*) and mean squared error (*MSE*):

$$ME = \frac{\sum_{e=1..n}(l_e)}{n} \quad (17)$$

$$MSE = \frac{\sum_{e=1..n}(l_e^2)}{n} \quad (18)$$

with n, the number of events, where $l_e = \log_{10}(d_s+0.1)-\log_{10}(d_m+0.1)$, with d_s, the simulated cumulative discharge at the outflow point (m^3), and d_m the measured cumulative discharge (m^3). The *PIntVar* scenario (Table 5.6, Figure 5.11) gives better results regarding *MSE* than all other scenarios, indicating that this scenario gives a better representation of the infiltration process on the hillslope than the other scenarios. Using the average of the saturated conductivity derived from field measurements of infiltration does not yield runoff, as shown by the *Ring* and *Plot* scenarios. The *PIntEvent* scenario results in an underestimation of discharge for most events. This can be explained by the transient conditions in the phases of the rainstorm corresponding to the rising and falling limb of the hydrograph. The *PIntVar* scenario deals with the transient conditions by using a temporally variable value for k (Figure 5.12) resulting in a smaller value of *MSE*. For analyzing unexplained variation in cumulative discharge, a multiple linear

Table 5.5. Summary statistics of the rain storms. n, number of events; p_{cum}, cumulative rain (mm) over event; p, average rain intensity during measured runoff; p_{cv}, coefficient of variation of rain intensity values (values are average values over periods of 30 s.), during measured runoff; d_Q, duration of measured runoff; d_{tt}, average travel time of water from catchment boundaries estimated with the rainfall runoff model. Values in brackets denote standard deviations.`

n	p_{cum} (mm)	p (mm/h)	p_{cv} (%)	d_Q (h)	d_{tt} (h)
16	19 (9.9)	8.4 (8.2)	91 (60)	1.8 (1.8)	0.22 (0.039)

Table 5.6. Mean error (*ME*) and mean squared error (*MSE*) of cumulative discharge for scenarios described in text. A '-' indicates that the model predicts zero discharge for all events.

Scenario	ME	MSE
PIntEvent	-0.8	1.0
PIntVar	-0.4	0.4
Ring	-	-
Plot	-	-
InvMod	0.4	0.8

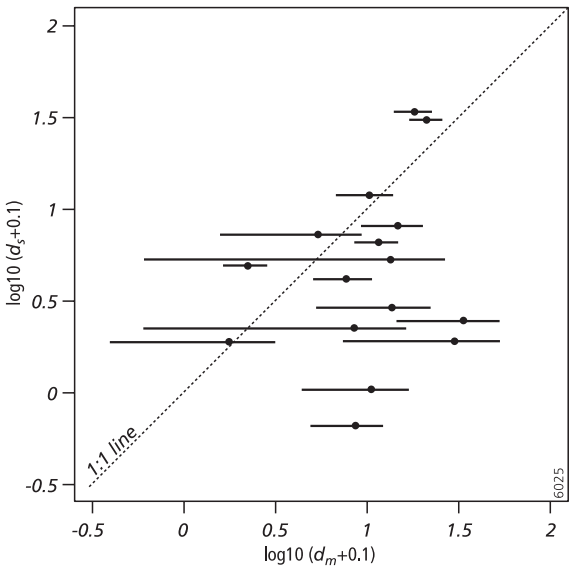


Figure 5.11. *PIntVar* scenario, $\log_{10}$ of simulated cumulative discharge (d_s, m^3) against estimated measured cumulative discharge (d_m, m^3). Each dot with a horizontal line represents one rainstorm, where the dot gives the cumulative discharge and the horizontal lines indicate measurement error, with endpoints of each line representing $d_{m,min}$ and $d_{m,max}$ of the measured discharge.

regression between l_e for the *PIntVar* scenario and (log transformed) characteristics of the events in Table 5.5 was performed. A backward elimination procedure using the Akaike information criterion as indicator for dropping terms, resulted in a significant positive relation between l_e and 1) rain intensity during runoff (p), and 2) average travel time of water from catchment boundaries (d_{tt}), with an r^2 of 0.71. Since both p and d_{tt} are related to the estimation of k in this scenario, the regression indicates that the approach followed by the *PIntVar* scenario is still not completely capable of dealing with the transient phases of runoff. Other approaches using a variable k during the event were tested, but these did not result a decrease of the *MSE*.

Although the use of the p-median(K_e^*) curve improves model results compared to scenarios with a constant k, the method applied in the *PIntVar* scenario still yields poor results for the transient case. This is mainly caused by an incorrect representation of the infiltration in the transient phases represented by the rising and falling limb of the hydrograph. Although it is not proven in this study, it can be expected that this limitation is a smaller problem for transient simulations of smaller hillslopes, having shorter travel times, resulting in a shorter duration of highly transient phases. This is indicated by the study of *Yu et al.* (1997). Following a similar approach of using a p-K_e^* curve in simulations of runoff from large plots (20-216 m^2), they found a close fit between modeled and measured hydrographs resulting from natural rainstorms. For larger hillslopes, it is expected that a scaling approach similar to the approach followed here can only be used in transient simulations when effective values of saturated conductivity are calculated for grid cells.

Apart from the above mentioned errors in the conceptualization of the processes, other factors may have caused part of the difference in measured and simulated discharge on the hillslope. These are 1) uncertainty in the probability distribution of $K_u(\mathbf{s})$ due to the relatively small number of ring infiltrometer experiments, 2) incorrect calibration of the

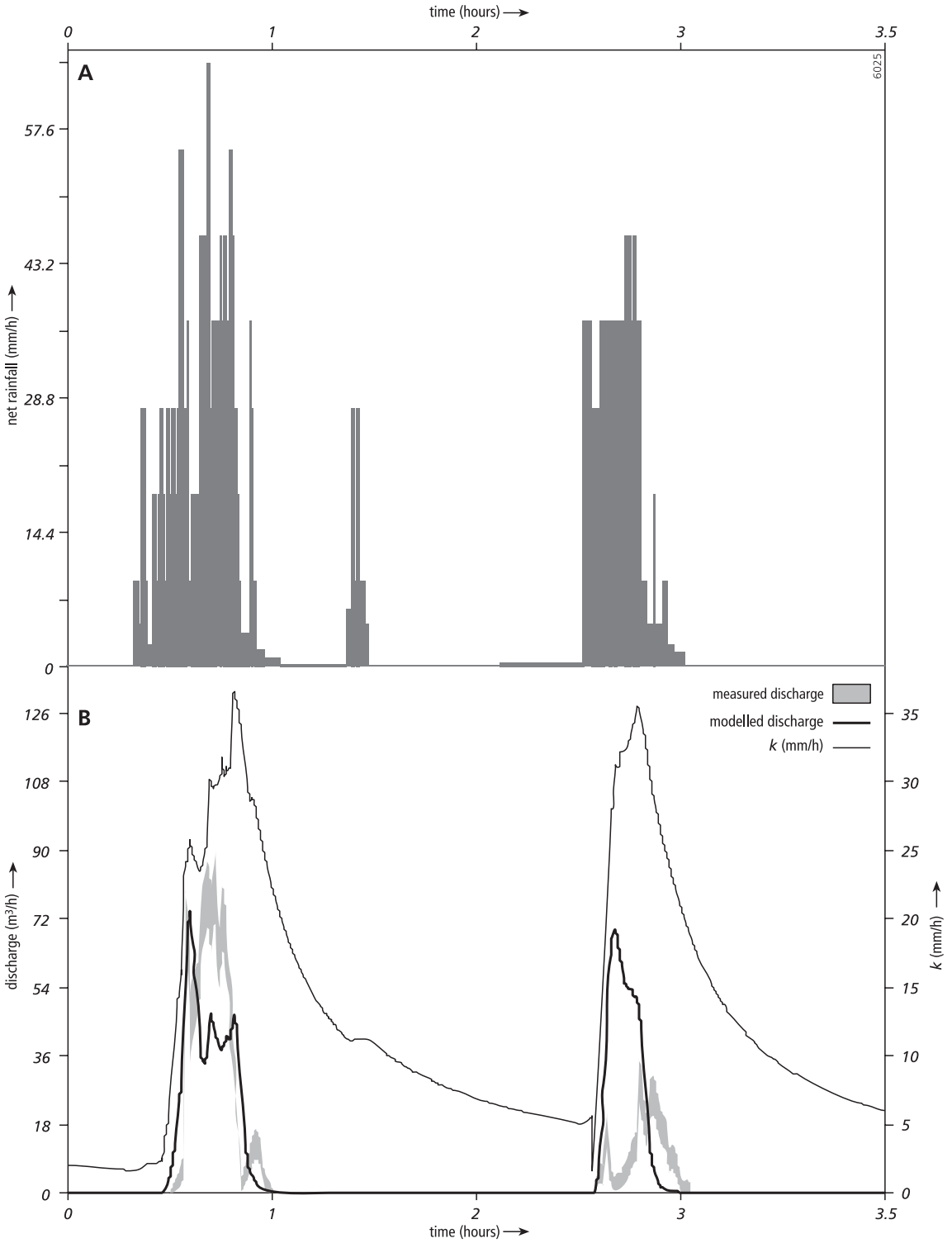


Figure 5.12. *PIntVar* scenario, event 6 Nov. 1997. (A) rainfall, (B) measured discharge, modelled discharge, and saturated conductivity ($k(t)$). Gray area of measured discharge represents error in measurements.

semivariogram range parameter *a* as a result of a relatively large number of rainfall simulation data at rain intensities higher than these of natural rainstorms, 3) occurrence of processes not incorporated in the model, such as removal of crusts at higher rain intensities and increased infiltration rates at the start of a rainstorm as a results of suction forces in the soil under unsaturated conditions.

5.6 Conclusions

Under steady state conditions, the scaling model predicts an increase in effective saturated conductivity of a model domain with 1) an increase in rain intensity, 2) increasing mean and decreasing variance, and skewness of saturated conductivity distribution within the model domain, 3) decreasing spatial scale of variation in saturated conductivity within the model domain, 4) increasing size of the model domain and, 5) decreasing bifurcation of the drainage pattern in the model domain. The direction of these relations corresponds with these found in other modeling studies and estimates of effective saturated conductivity derived from field measurements in other studies.

In addition, it has been shown that the scaling model is capable of scaling modeled saturated conductivity from the local scale (0.04 m^2) to the plot scale (1 m^2). Estimates of saturated conductivity at the local scale derived from ring infiltrometer measurements representing the local scale values can be scaled to effective values at the plot scale corresponding to these derived from rainfall simulation experiments at that scale.

At the hillslope scale (7500 m^2), the relation between rain intensity and effective saturated conductivity found by the scaling model was tried in a transient simulation of runoff from the hillslope. Results were generally poor, most probably owing to the steady state nature of the effective conductivity relationship. However, effective values did at least create runoff, which was not the case when directly using the average saturated conductivity derived with the field measurements of infiltration. Moreover, using an approach with varying effective saturated conductivity with rainfall intensity gave better results than using a fixed saturated conductivity per event, or a value obtained from inverse modeling. This indicates that an approach with varying effective saturated conductivity with rainfall intensity does have some potential in transient runoff modeling. It is expected that transient simulations will give better results when smaller areas are used as modeling domain, for instance grids.

Acknowledgements
Simone van Dijck is gratefully acknowledged for inputs, helpful comments on earlier drafts, and providing the dataset. I am also grateful to Marc Bierkens, Peter A. Burrough and Willem van Deursen for comments and inputs to the work. Many thanks to members of the PCRaster team in Utrecht (Kor de Jong and Cees Wesseling) who provided technical support during the project. Jakolien Leenders and Judith Snepvangers are thanked for providing their ring infiltrometer data set. Christian Salles, Laboratory for Experimental Geomorphology of the Catholic University of Leuven, Belgium, provided support in measuring kinetic energy of artificial rainfall. Ton Markus is thanked for drawing the figures.

5.7 References

Aitchison, J. & J.A.C. Brown (1957), The lognormal distribution, with special reference to its uses in economics: Cambridge University Press, Cambridge.

Battany, M.C. & M.E. Grismer (2000), Development of a portable field rainfall simulator for use in hillside vineyard runoff and erosion studies, Hydrological Processes 14, pp. 1119-1129.

Beven, K. (1989), Changing ideas in hydrology - the case of physically-based models, Journal of Hydrology 105, pp. 157-172.

Bierkens, M.F.P., P.A. Finke & P. de Willigen (2000), Upscaling and downscaling methods for environmental research, Kluwer Academic Publishers, Dordrecht, Boston.

Binley, A. & K. Beven (1989), A physically based model of heterogeneous hillslopes: 2. Effective Hydraulic Conductivities, Water Resources Research 25, pp. 1227-1233.

Blöschl, G. (1996), Scale and scaling in hydrology: Wiener Mitteilungen 132, Technische Universität Wien, Institut für Hydraulik, Gewässerkunde und Wasserwirtschaft, Wien.

Blöschl G, R.B. Grayson & M. Sivapalan (1995), On the representative elementary area (REA) concept and its utility for distributed rainfall-runoff modelling, Hydrological Processes 9, pp. 313-330.

Blöschl, G. & M. Sivapalan (1995), Scale issues in hydrological modelling: a review. Hydrological Processes 9, pp. 251-290.

Burrough, P.A. & R.A. McDonnell (1998), Principles of Geographical Information Systems, Oxford University Press, Oxford.

Burt, T.P. (1998), Infiltration for soil erosion models: some temporal and spatial complications. In, Modelling Soil Erosion by Water. Eds J. Boardman and D. Favis-Mortlock. NATO ASI Series, Vol I 55.

Buttle, J.M. and D.A. House (1997), Spatial variability of saturated hydraulic conductivity in shallow macroporous soils in a forested basin, Journal of Hydrology 203, pp. 127-142, 1997.

Chow, V.T., D.R. Maidment & L.W. Mays (in press), Applied Hydrology, McGraw-Hill, NY, 1988.

Corradini, C., R.S. Govindaraju, R. Morbidelli (in press), Simplified modelling of areal average infiltration at the hillslope scale. Hydrological Processes.

Corradini, C., R. Morbidelli & F. Melone (1998), On the interaction between infiltration and Hortonian runoff, Journal of Hydrology 204, pp. 52-67.

De Roo, A.P.J., C.G. Wesseling, C.J. Ritsema (1996), LISEM: a single-event physically based hydrological and soil erosion model for drainage basins. I: theory, input and output. Hydrological Processes 10, pp. 1107-1117, 1996.

Grant, S.A., J.D. Jabro, D.D. Fritton & D.E. Baker (1991), A stochastic model of infiltration which simulates “macropore” soil water flow, Water Resources Research 27, pp. 1439-1446, 1991.

Hammersley, J.M. & D.C. Handscomb, 1979, Monte Carlo Methods, Chapman & Hall, London, 1979.

Harms, T.E. & D.S. Chanasyk (2000), Plot and small-watershed scale runoff from two reclaimed surface-mined watersheds in Alberta, Hydrological Processes 14, pp. 1327-1339, 2000.

Hendrayanto K.K. & T. Mizuyama (2000), Scaling hydraulic properties of forest soils, Hydrological Processes 14, pp. 521-538, 2000.

Heuvelink, G.B.M. (1998), Error Propagation in Environmental Modelling with GIS, London, Taylor & Francis, London, 1998.

Joel, A., I. Messing, O. Seguel, M. Casanova (in press), Measurement of surface water runoff from plots of two different sizes. Hydrological Processes.

Kamphorst, E.C., V. Jetten, J. Guérif, J. Pitkänen, B.V. Iversen, J.T. Douglas & A. Paz (2000), Predictiong Depressional Storage from Soil Surface Roughness, Soil Sci. Soc. of Am. J. 64, pp. 1749-1758, 2000.

Knisel, W.G., Williams, J.R. (1995), Hydrology components of CREAMS and GLEAMS models. In: V.P. Singh, Computer models of watershed hydrology. Water Resources Publications, 1995, pp. 1069-1114.

Lauren, J.G., R.J. Wagenet, J. Bouma & J.H.M. Wosten (1988), Variability of saturated hydraulic conductiviety in a glossaquic hapludalf with macropores, Soil Science 145, pp. 20-28.

Li, R.-M., D.B. Simons & M.A. Stevens (1975), Nonlinear kinematic wave approximation for water routing, Water Resources Research 11, pp. 245-252.

Loage, K., G.A. Gander (1990), R-5 Revisited. 1. Spatial Variability of Infiltration on a Small Rangeland Catchment. Water Resources Research 26, pp. 957-971.

Loague, K., P.C. Kyriakidis (1997), Spatial and temporal variability in the R-5 infiltration data set: Déjà vu and rainfall-runoff simulations. Water Resources Research 33, pp. 2883-2895.

Mallants, D., B.P. Mohanty, A. Vervoort & J. Feyen (1997), Spatial analysis of saturated hydraulic conductivity in a soil with macropores, Soil Techn. 10, pp. 115-131.

Merriam, R.A. (1973), Fog drip from artificial leaves in a fog wind tunnel, Water Resources Research 9, pp. 1591-1598.

Merz, B., A. Bárdossy, G.R. Schiffler (in press), Different methods for modelling the areal infiltration of a grass field under heavy precipitation. Hydrological Processes.

Merz, B. & E.J. Plate (1997), An analysis of the effects of spatial variability of soil and soil moisture on runoff, Water Resources Research 33, pp. 2909-2922.

Morgan, R.P.C., J.N. Quinton, R.E. Smith, G. Govers, J.W.A. Poesen, K. Auerswald, G. Chisci, D. Torri & M.E. Styczen (1998), The european soil erosion model (EUROSEM): a dynamic approach for predicting sediment transport from fields and small catchments. Earth Surface Processes and Landforms 23, pp. 527-544.

Orlandini S., A. Perotti, G. Sfondrini & A. Bianchi (1999), On the storm flow response of upland Alpine catchments. Hydrological Processes 13, pp. 549-562.

Pebesma, E.J. & G.B.M. Heuvelink (1999), Latin hypercube sampling of Gaussian random fields. Technometrics 41, pp. 303-312.

Philip, J.R. (1987), The Infiltration Joining Problem, Water Resources Research 23, pp. 2239-2245.

Philip, J.R. (1969), Theory of infiltration, Adv. Hydrosci., 5, pp. 215-296, 1969.

Ragab, R. & J.D. Cooper (1993), Variability of unsaturated zone water transport parameters: implications for hydrological modelling. 2. Predicted vs. in situ measurements and evaluation of methods. Journal Hydrology 148, pp. 133-147.

Russo, D. (1992), Upscaling of Hydraulic conductivity in Partially Saturated Heterogeneous Formation. Water Resources Research 28, pp. 397-409.

Russo, D. & E. Bresler (1981), Soil Hydraulic Properties as Stochastic Processes: I. An Analysis of Field Spatial Variability, Soil. Sci. Soc. Am. J., 45, pp. 682-687.

Salles, C. & J. Poesen (1999), Performance of an Optical Spectro Pluviometer in measuring basic rain erosivity characteristics, Journal Hydrology, 218, pp. 142-156.

Sharma, M.L., G.A. Gander & C.G. Hunt (1980), Spatial variability of infiltration in a watershed, Journal Hydrology 45, pp. 101-122.

Smettem, K.R.J. (1987), Characterization of water entry into a soil with a contrasting textural class: spatial variability of infiltration parameters and influence of macroporosity, Soil. Sc. 144, pp. 167-174.

Smith, R.E., D.C. Goodrich, D.A. Woolhiser, C.L. Unkrich (1995), KINEROS – A kinematic runoff and erosion model. In: V.P. Singh, Computer models of watershed hydrology. Water Resources Publications, 1995, pp. 697-732.

Smith, R.E. & R.H.B. Hebbert (1979), A Monte Carlo analysis of the hydrologic effects of spatial variability of infiltration, Water Resources Research 15 (2), pp. 419-429.
Smith, R.E. & Parlange, J.-Y. (1978), A parameter-efficient hydrologic infiltration model. Water Resources Research 14, pp. 533-538.
Sullivan, M., J.J. Warwick & S.W. Tyler (1996), Quantifying and delineating spatial variations of surface infiltration in a small watershed, Journal of Hydrology 181, pp. 149-168.
Touma J. & J. Albergel (1992), Determining soil hydrologic properties from rain simulator or double ring infiltrometer experiments: a comparison, Jounal Hydrology 135, pp. 73-86.
Van de Giesen, N.C., T.J. Stomph & N. de Ridder (2000), Scale effects of Hortonian overland flow and rainfall-runoff dynamics in a West African catena landscape, Hydrological Processes 14, pp. 165-175.
Van Dijck, S.J.E. (2000), Effects of agricultural land use on surface runoff and erosion in a Mediterranean area, Ph.D. thesis, Utrecht University, NGS Publication 265, 246 pp.
Wesseling, C.G., D. Karssenberg, W.P.A. van Deursen & P.A. Burrough (1996), Integrating dynamic environmental models in GIS: the development of a Dynamic Modelling language. Transactions in GIS 1, pp. 40-48.
Williams, J. and M. Bonell (1988), The influence of scale of measurement on the spatial and temporal variability of the Philipp infiltration parameters − an experimental study in an australian savannah woodland, Journal of Hydrology 104, pp. 33-51.
Woolisher, D.A., R.E. Smith & D.C. Goodrich (1990), KINEROS, a kinematic runoff and erosion model: documentation and user manual, 130 p.
Woolhiser, D.A., R.E. Smith & J.-V. Giraldez (1996), Effects of spatial variability of saturated hydraulic conductivity on Hortonian overland flow, Water Resources Research 32, pp. 671-678.
Yu, B., C.W. Rose, K.J. Coughlan, B. Fentie (1997), Plot-scale rainfall-runoff characteristics and modeling at six sites in Austraila and Southeast Asia. Transactions of the American Society of Agricultural Engineers 40, pp. 1295-1303.

6 A SCALE TRANSFORM FUNCTION TO COMPUTE SATURATED CONDUCTIVITY FOR MODEL UNITS OF DYNAMIC SPATIAL RAINFALL-RUNOFF MODELS FROM LOCAL SCALE MEASUREMENTS OF INFILTRATION

D. Karssenberg

Abstract: Dynamic spatial rainfall-runoff models require values of saturated conductivity which are representative for (i.e., effective at) the support size used in these models. The support size is defined as the domain in space and time in which all fluxes are assumed to be homogeneous and constant, respectively. It is defined by the size of the model units and the time step used in the rainfall-runoff model. This chapter describes a scale transform function to upscale local scale values of saturated conductivity to effective values of saturated conductivity at a support corresponding to the size of a model unit of 100 m^2 and a time step of several seconds. The form of the transfer function is derived from a stochastic model simulating the spatial process of runoff and infiltration within a model unit. The three parameters in the transfer function can be found for each model unit in the rainfall-runoff model using a correlation between these parameters and 1) the spatial probability distribution of the saturated conductivity within a model unit, 2) derivatives of the digital elevation model at the scale of the model unit (e.g., slope, curvature), and 3) the pattern of surface runoff in the unit. With field data from a 0.43 km^2 catchment in the Ouvèze river basin, S. France, the transfer function was tested in a simulation with a dynamic spatial rainfall-runoff model, resulting in an effective saturated conductivity that varied for each model unit with the intensity of net rainfall and runon to the model unit. The simulations showed that the application of the transfer function in a dynamic spatial rainfall-runoff model resulted in differences between simulated and measured cumulative discharge from the catchment which were smaller than these found with simulations without the use of the transfer function. This was also the case for a small hillslope in the same area.

6.1 Introduction

When it is simulated over a small area (e.g. 10^{-2} m^2), infiltration is generally considered to be a one dimensional process. A large number of standard infiltration models are available to do such simulations. In most cases these models estimate infiltration as a function of the availability of surface water for infiltration, physical and chemical properties of the soil, water content of the soil, occurrence of macro pores and/or surface sealing. For larger areas, infiltration has to be considered as a spatial process with an interaction between overland flow and infiltration processes in the soil, since the availability of water for infiltration becomes dependent on rain and runon, while runon is determined by the same interaction process in upstream areas. As a result, for larger areas, infiltration can only be modeled when both infiltration processes in the soil and runoff are regarded as interrelated processes with variation in space (*Smith and Hebbert*,

1979; *Woolisher et al.*, 1996; *Merz and Plate*, 1997; *Corradini et al.*, 1998). In a spatial infiltration model, this means that factors determining one dimensional infiltration need to be incorporated, but also the spatial variation in these factors and the spatial pattern of runoff.

Infiltration as a spatial process can be simulated by dynamic spatial rainfall-runoff models simulating discharge from a catchment, using model units with such a small size that within each unit infiltration can be simulated as a one dimensional process into the soil, ignoring spatial processes within the model unit. Since spatial variation in runoff and infiltration occurs at very small scales, this approach leads to rainfall-runoff models with model units that need to be very small (e.g. 10^{-2} m^2), resulting in unacceptably large model run times. Moreover, field data needed as model input, for instance to define the flow pattern between model units, is mostly not available at this high resolution. The only alternative, especially for large catchments, is to use larger model units. It is common practice in dynamic rainfall-runoff modeling to use model units with an area of 10^1-10^8 m^2. Data at this resolution are usually derived from existing digital elevation models, digitized soil, vegetation or land use maps or remote sensing images.

As noted above, infiltration cannot be regarded as a one dimensional process when larger model units are used. Thus, so called effective parameters have to be defined for model units; i.e. parameters that take account of the infiltration and runoff-runon processes within each model unit. In a dynamic spatial model, an effective parameter or variable is the single value assigned to a model unit, defined in the space and time domain, such that the model based on that value yields the same output for that model unit as a model based on the actually occurring heterogeneous parameter or variable field(s) within the model unit (after *Blöschl and Sivapalan*, 1995). The optimal way to estimate effective parameters for infiltration would be to adjust the support of field measurements of infiltration to the size of the model units used. Although possible, this has unacceptable disadvantages since it results in large infiltration plots which are difficult to install. Moreover, infiltration plots ignore runon from upstream areas outside the plot. In most cases, the only measurements available are those estimating infiltration as a one dimensional process into the soil, such as ring infiltrometer measurements or laboratory measurements of conductivity in soil samples, or measurements with a rainfall simulator measuring infiltration as a spatial process at a small support (typically 1-10^2 m^2). These measurements can only be used for modeling infiltration in a dynamic spatial rainfall-runoff model using a mathematical or numerical upscaling technique.

Although many authors have stressed the need for upscaling techniques (*Beven*, 1989; *Binley et al.*, 1989; *Blöschl et al.*, 1995; *Blöschl and Sivapalan*, 1995; *Harms and Chansyk*, 2000; *Bierkens et al.*, 2000), research in this field is in its infancy. Existing methods for deriving effective parameters (e.g., *Russo*, 1992; *Hendrayanto et al.*, 2000) mostly ignore the interaction between surface runoff and infiltration. Studies that take the interaction between runoff and infiltration into account are either focused on effective parameters for (sub-)catchments as a whole (e.g., *Binley and Beven*, 1989), or do not deal with infiltration under spatial variation of runoff (e.g., *Grant et al.*, 1991). To the author's knowledge, upscaling techniques for deriving effective infiltration parameters for model units with inflow from upstream are not available.

This study describes an upscaling method that calculates effective values of saturated conductivity for model units of dynamic spatial rainfall-runoff models. These effective

values are derived from field measurements of saturated conductivity with ring infiltrometers or rainfall simulators. The technique takes into account the interaction between runoff and infiltration, in an approach similar to the approach described in chapter 5. It is developed for a catchment with macro pore flow dominated infiltration, assuming a temporally constant spatial pattern of saturated conductivity within the model unit. The aim is to develop and test a transfer function which derives the effective saturated conductivity of a model unit from the mean of the effective saturated conductivity field within the unit and additional parameters. These additional parameters can be estimated from data which are available in most rainfall-runoff modeling studies: parameters describing the probability distribution of the effective saturated conductivity, the derivatives of the digital elevation model used (e.g., slope, curvature), and the pattern of surface flow within the unit (e.g., sheet flow or rill flow).

The setup of the chapter is as follows. First, a description of the study area is given, followed by a general overview of the approach followed. Then, a steady state rainfall-runoff model is described simulating the spatially variable process of infiltration within a model unit. From the concepts and outcomes of this stochastic model, the transfer function is developed, which is applied in a dynamic spatial rainfall-runoff model simulating discharge from two catchments. The results are evaluated by comparing simulated and measured discharge, and by additional runs with the rainfall-runoff model without the use of the transfer function.

6.2 Study area and methods

The study area corresponds to the study area described in Chapter 5, comprising the 'La Folia' catchment in the Ouvèze river basin, S. France, which is a 0.43 km^2 cultivated catchment consisting of approximately 100 arable fields, mainly vineyards, near the village of Entrechaux. It contains loamy sands, sandy loams, loams and silt loam soils (USDA classification), all thicker than 2 m over bedrock, with hillslope angles of 0.25 m/m on average. The catchment has a sub-Mediterranean climate, with a mean annual rainfall of 800 mm. This catchment is referred to here as catchment A. It contains catchment B, the catchment for which dynamic simulations were done in Chapter 5, which is an approximately rectangular 7500 m^2 sandy loam hillslope with a vineyard. Its mean slope is 0.042 m/m and the constant direction of tillage on the hillslope is approximately parallel to the elevation contour lines.

Eighteen infiltration measurements in catchment A were made with double-ring infiltrometers in May and June, 1997. The inner ring was 20 cm in diameter and the outer one was 40 cm in diameter. Rings were driven 3-5 cm into the ground and smeared with cement on the side, to minimize leakage. In addition, rainfall simulation experiments were done on 0.02 m^2-5.5 m^2 plots at 173 randomly chosen locations on cultivated land in catchment A, in the period May-July, 1994-1997. With these field measurements of infiltration, the research described in Chapter 5 derived the spatial probability distribution of the saturated conductivity $K_u(\mathbf{s})$ of the top-soil for catchment A, for a support of rectangular cells of 0.04 m^2 area, which was calculated to have a lognormal distribution, with a mean $m_{K_u(\mathbf{s})}$ = 1004 mm/h, a variance of $\ln(K_u+1)$ of 7.73, and a spatial pattern of

$\ln(K_u+1)$ characterised by a spherical variogram with a range a_{var} = 1.2 m. The rainfall simulation experiments did not reveal significant differences between soil and landuse types in catchment A. The high mean value of the saturated conductivity is probably caused by macro pore flow (c.f., chapter 5), while surface crusting was shown not to have a significant impact on the infiltration rate.

Field data to derive model input parameters and variables for rainfall-runoff modeling were collected in catchment A and B, in 1997 and 1998 (Table 6.1, c.f. *Van Dijck*, 2000). Hydrographs and the cumulative discharge (d_m, m^3) under Hortonian runoff from catchment A (n = 16 events) and B (n = 14 events) were derived from measurements with flumes installed with continuous pressure recorders. Errors in the water depth and velocity head measurements recorded by the pressure recorders were estimated resulting in minimum ($d_{m,min}$, m^3) and maximum ($d_{m,max}$, m^3) values of cumulative discharge.

Table 6.1. Source of parameter values for running the rainfall-runoff model on catchment A and B. All data collected on catchment A, c.f., *Van Dijck* (2000). For rainfall simulations and ring infiltrometer measurements, see text.

Model component	Parameter	Source	N
Catchment representation and flow direction	digital elevation model	interpolation from elevation contours	-
	tillage direction	field observations	all fields
Rain	rain intensity (p_r)	tipping bucket, Casella, 0.077 mm/tip	1 location
Interception	vegetation cover (c)	field measurement of canopy area and vertical photographs of canopy	Several in winter and summer
	maximum content of interception store (s_m)	throughfall measurements after rainstorms	20
Surface storage	maximum surface storage	based on field measurements and Kamphorst *et al.* (2000)	-
Surface runoff	discharge	flume with continuous pressure recorders	1 per catchment
	manning's n	Chow (1959)	-

* Centre d'Information Régional Agrométéorologique et Economique, Carpentras

6.3 Approach

The measured discharge from catchment A and B is simulated with a dynamic spatial rainfall-runoff model with model units of 10 x 10 m. The aim is to develop and parameterize a transfer function calculating the effective saturated conductivity for each model unit in the rainfall-runoff model, and to apply the function in this model, to both catchment A and B. The approach is as follows.

Dynamic spatial rainfall-runoff models use model units which are subdomains of the spatial and temporal domain, created by discretisation of the spatial domain in grid cells or polygons (Figure 6.1A), and discretisation of the time domain in time slices. For each model unit, an effective saturated conductivity ($k_e(t)$, mm/h) is needed, where t represents the time (s). The value of $k_e(t)$ of a unit is related to several properties of the unit:

$$k_e(t) \sim (p(t), q(t), K_u(\mathbf{s}), \textit{flow pattern}, \textit{location}) \quad (1)$$

Since $k_e(t)$ depends on the availability of water for infiltration (chapter 5), it will depend on the net rain reaching the surface ($p(t)$, mm/h) and the inflow ($q(t)$, mm/h) from neighboring units (i.e., subdomains), both averaged over the unit. The fluxes $p(t)$ and $q(t)$ can be derived from the rainfall-interception module and the water routing algorithm, respectively, of the rainfall-runoff model in which $k_e(t)$ is needed. Since the spatial pattern of the saturated conductivity is not exactly known for each model unit, it is modeled with a random field $K_u(\mathbf{s})$ with known spatial probability distribution. In addition to $p(t)$, $q(t)$, and $K_u(\mathbf{s})$ which have an effect on $k_e(t)$, the spatial pattern of surface flow within the unit is important, since it has effect on the interaction between runoff and infiltration. Just like the spatial pattern of infiltration, the flow pattern is not exactly

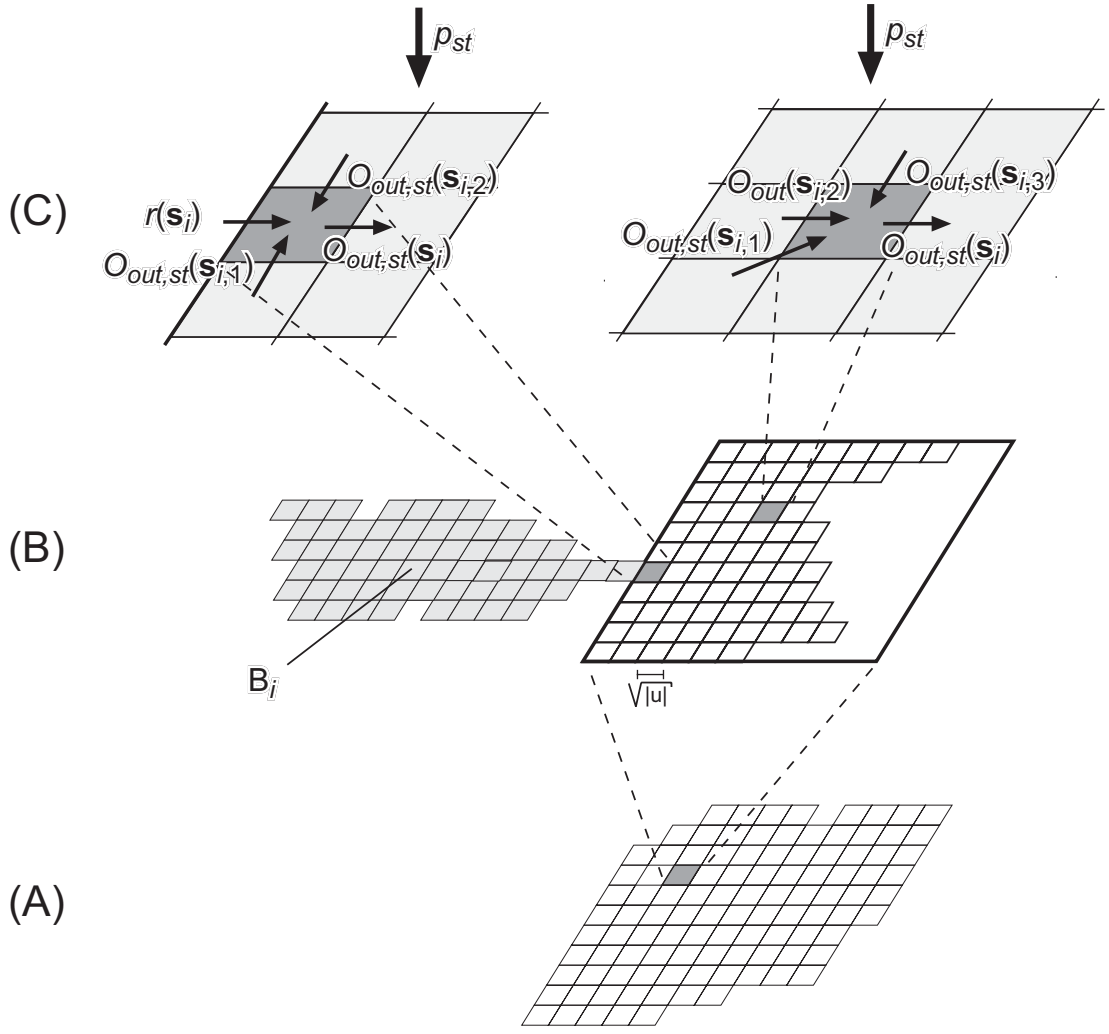


Figure 6.1. (A) Model units in a rainfall-runoff model using rectangular model units; (B) model unit discretised in subunits with area |u| (m^2), catchment area B_i shown of one subunit at the edge of the unit; (C) input and output fluxes for one subunit within a unit; left, subunit at the edge of the unit; right: subunit not at the edge of the unit.

known for each unit, and it needs to be represented as a random variable, with known statistical properties. The location of the unit in the large scale drainage network will influence the value of $k_e(t)$ since it determines whether runon to the unit is concentrated at a few locations or spread over the cell.

For describing the process of runoff and infiltration occurring within the unit, a steady-state stochastic spatial model (G) is developed for within unit runoff-infiltration modeling. Under steady state conditions, it derives the probability distribution of the effective saturated conductivity for each unit, using the inputs in equation 1. This stochastic model cannot be used for calculating the effective saturated conductivity distribution for each unit in a rainfall-runoff model with a large number of units, since its run time is very large. For this reason, a deterministic transfer function g will be derived from the properties and outcomes of the steady-state stochastic model G. It transfers the mean saturated conductivity ($m_{K_u(s)}$) of the unit to the effective saturated conductivity ($k_e(t)$) of the unit. Under the assumption that steady state conditions occur over each time interval (t, $t+\Delta t$), this transfer function can be used to calculate $k_e(t)$:

$$k_e(t) = g\left[p(t), q(t), p_1, p_2, \ldots p_n\right] m_{K_u(s)} \qquad (2)$$

The variables $p(t)$ and $q(t)$ are input variables to the transfer function, for each time step in the rainfall-runoff model, p_1, p_2,..p_n are parameters, which are related to the spatial probability distribution of the saturated conductivity, the flow pattern, and the location of the unit (equation 1). By running the stochastic model G for within unit runoff-infiltration modeling for each unit in catchment B, which is possible, since it contains only 75 units, the parameter values $p_{1..n}$ in the transfer function g will be derived for each unit in this catchment, by fitting the transfer function g to the outcome of the stochastic model G.

The same approach for finding the parameter values $p_{1..n}$ for all units in the larger catchment A cannot be applied, since this would take too much calculation time. Instead a regression analysis is applied using the results of catchment B. Since the parameters p_1, p_2,...p_n are related to (known) properties of the cell (equation 1), a regression will be performed for catchment B between these parameters as dependent variables and 1) parameters describing the spatial probability distribution of the saturated conductivity, 2) the flow pattern and 3) the location of the unit in the large scale drainage system as independent variables. The results of this regression can be used for estimating the parameter values p_1, p_2,...p_n for individual model units in catchment A.

Finally, the dynamic spatial rainfall-runoff model with the transfer function g and its estimated parameter values is used to simulate the discharge of catchment A and B for all rainstorms. Results of this simulation will be compared with field measurements of discharge and results of simulations without the transfer function.

6.4 Stochastic model for within-unit runoff-infiltration modeling

6.4.1 Stochastic representation of within unit saturated conductivity and flow pattern

A model unit with spatial variation in saturated conductivity is represented by the random field (i.e. random function) $\{K(\mathbf{s}) : \mathbf{s} \in D\}$, where $\mathbf{s}$ is a spatial coordinate in the two dimensional domain D. Note that uppercase letters denote random fields (e.g., K), while lower case letters denote realizations of random fields (e.g., k). In a model unit, flow over the surface and infiltration are assumed to occur over square subunits u($\mathbf{s}$), with an area |u| (m^2, Figure 6.1B). Since |u| is chosen to be very small, typically 0.04 m^2, it is assumed that rain on a subunit and inflow from upstream to the subunit are distributed evenly over the whole subunit. Under this assumption, the saturated conductivity at the support of a subunit $K_u(\mathbf{s})$ is the average of $K(\boldsymbol{s})$ over the subunit u:

$$K_u(\mathbf{s}) = \frac{\int_u K(\mathbf{s})\mathrm{d}\mathbf{s}}{|u|} \qquad (3)$$

At the support |u|, the plausible assumption can be made that $K_u(\mathbf{s})$ has a lognormal distribution (*Russo and Bresler*, 1981; *Ragab and Cooper*, 1993; *Buttle and House*, 1997; *Mallants et al.*, 1997):

$$K_u(\mathbf{s}) = e^{Z(\mathbf{s})}. \qquad (4)$$

Furthermore, it is assumed that $Z(\mathbf{s})$ is a multivariable normal and stationary random spatial function. Thus, $\{Z(\mathbf{s}) : \mathbf{s} \in D\}$ is defined by:

$$m_{Z(\mathbf{s})} = \mathrm{E}\{Z(\mathbf{s})\}, \qquad (5)$$

$$\gamma(\mathbf{h}) = \frac{1}{2}\mathrm{E}\left\{\left(Z(\mathbf{s}) - Z(\mathbf{s}+\mathbf{h})\right)^2\right\}, \qquad (6)$$

where E represents expectation. The quantity $\gamma(\mathbf{h})$ which is a function only of the separation vector $\mathbf{h}$, is called the semivariogram defining the spatial structure of $Z(\mathbf{s})$. In the absence of other information, a spherical model is assumed for the variogram:

$$\gamma(\mathbf{h}) = \begin{cases} \sigma_{Z(\mathbf{s})}\left(\frac{3}{2}\frac{|\mathbf{h}|}{a_{\mathrm{var}}} - \frac{1}{2}\frac{|\mathbf{h}|^3}{a_{\mathrm{var}}{}^3}\right) & \text{for } 0 \le |\mathbf{h}| \le a_{\mathrm{var}} \\ \sigma_{Z(\mathbf{s})} & \text{for } |\mathbf{h}| > a_{\mathrm{var}} \end{cases} \qquad (7)$$

with; a_{var}, range of the variogram defining the spatial scale of variation; $\sigma_{Z(\mathbf{s})}$, maximum value of the covariance. The expectation (mean) of $K_u(\mathbf{s})$ is (*Aitchison and Brown*, 1957):

$$\mathrm{E}\{K_u(\mathbf{s})\} = m_{K_u(\mathbf{s})} = e^{m_{Z(\mathbf{s})}+\frac{1}{2}\sigma_{Z(\mathbf{s})}}, \quad (8)$$

Increasing $\sigma_{Z(\mathbf{s})}$ results in a probability distribution of $K_u(\mathbf{s})$ with both a higher variance and a higher skewness.

Surface flow within the unit is represented by flow over a rectangular grid of subunits $u(\mathbf{s}_i)$, with $\mathbf{s}_1,\ldots,\mathbf{s}_n$, the locations of the center of the subunits at a regular grid, with a grid spacing corresponding to the length and width of the subunits ($\sqrt{|u|}$, m), Figure 6.1B). Realizations of $K_u(\mathbf{s}_i)$ are generated by sequential Gaussian simulation with Latin hypercube sampling (*Pebesma and Heuvelink*, 1999).

The model assumes that the exact flow pattern between subunits is not known, although general characteristics will be known, such as the large scale change in topography and the presence or absence of micro relief such as furrows. The model generates realizations of the flow pattern derived from a digital elevation model defined as a random field with characteristics representing the large scale topography as a deterministic surface and micro relief added as a random component with specified spatial correlation characteristics. For each realization, the model derives from such a realization of the digital elevation model the direction of flow for each subunit $u(\mathbf{s}_i)$, assigning to each subunit a flow direction to one of the eight steepest downstream neighboring units, while removing small, local depressions (8-point pour algorithm, *Burrough and McDonnell*, 1998).

To represent the flow patterns under different land use types in the study area, the flow pattern is generated for three scenarios with different types of random components added to the original, deterministic elevation model (Figure 6.2):

1. sheet flow. The drainage pattern is derived from the original digital elevation model plus a realization from a random field given by a spherical variogram with range = 1000 m and variance $5 \cdot 10^{-5}$ m.

2. flow with micro scale variation. The drainage pattern is derived from the original digital elevation model plus spatially correlated variation given by a spherical variogram with range = 0.5 m and variance 0.05 m.

3. flow with wheel tracks. The drainage pattern is derived from the original digital elevation model plus a realization of relief caused by wheel tracks. This wheel track relief consists of parallel, straight wheel tracks in a direction Γ (degrees), with a constant interval between the wheel track centers of 1.25 m, a wheel track width of 0.5 m, and a depth of 0.01 m. The direction Γ is a random variable with an average direction and a variance of 1. For each realization, a slightly different wheel track pattern was obtained by generating the wheel track pattern using the above given constants and a realization of Γ.

6.4.2 Process model

For modeling the factors determining the effective saturated conductivity of the model unit (Figure 6.1B), a process model (G) is used with the stochastic parameters described above. The aim of this model is to estimate the effective saturated conductivity of the unit over the time step Δt applied in the dynamic spatial rainfall-runoff model, for which the effective values are needed. Since model units and time slices are relatively small (typically 10^2 m^2 and 10 s, respectively), it is assumed that the effective saturated conductivity of the unit can be estimated with a steady state approach, using constant fluxes over Δt.

The process model is defined as follows. The subunits u($\mathbf{s}_i$) on a rectangular grid within the unit drain to each other. Any subunit within the unit draining to a neighboring

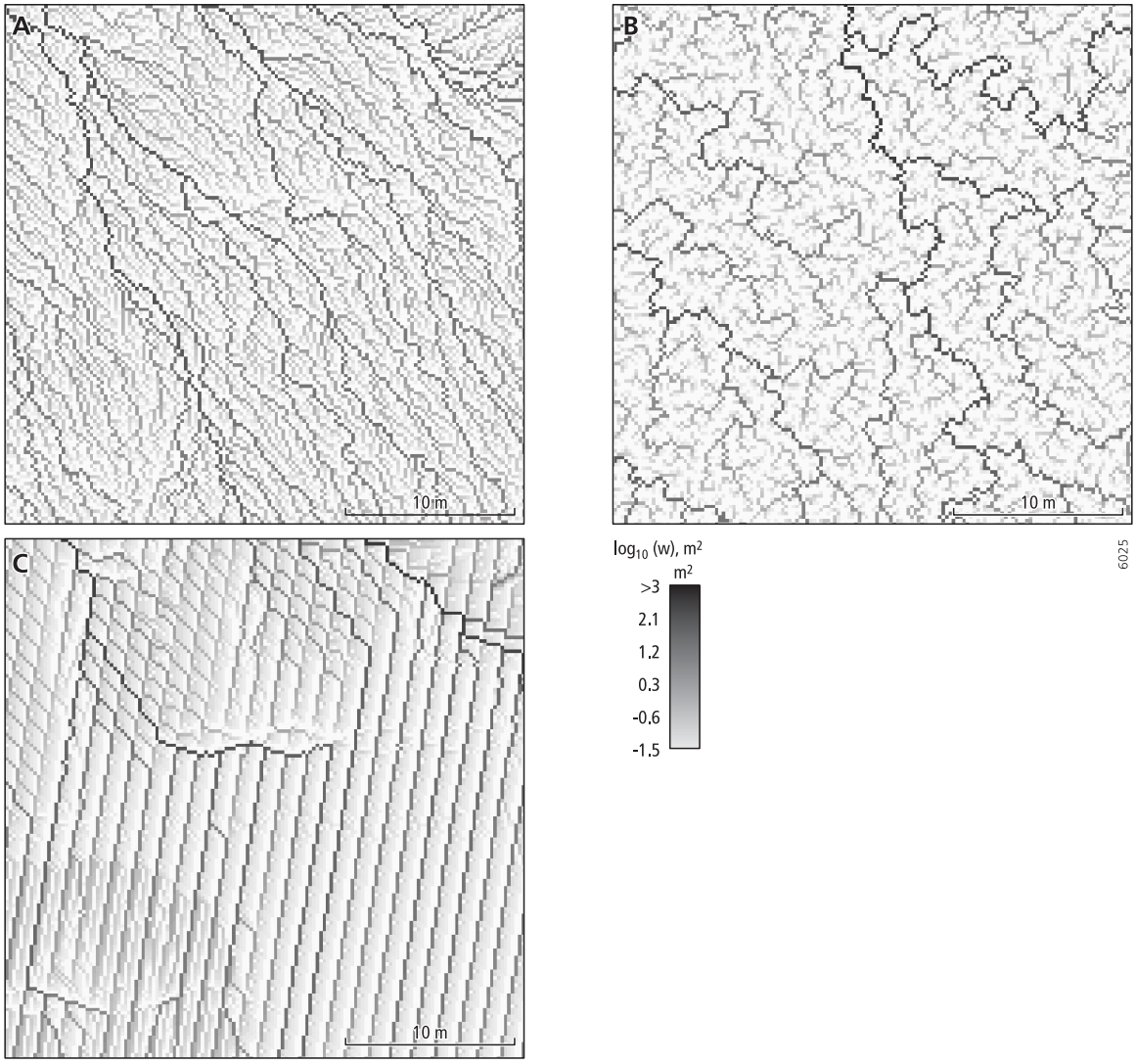


Figure 6.2. Example realizations of drainage patterns for a zoomed area of a field. Catchment area (w) of each subunit u($\mathbf{s}_i$). (A) sheet flow, (B) micro scale variation, (C) wheel tracks. Note that the length of one unit is 10 m.

subunit u($\mathbf{s}_i$) within the unit is called a neighboring upstream subunit u($\mathbf{s}_{i,n}$), n=1..m (Figure 6.1C), with m the number of neighboring subunits (max. 8). Since it is assumed that rainfall on the subunit and inflow from upstream to the subunit is evenly distributed over the whole subunit, as described in the section 6.4.1, the actual infiltration in a subunit is:

$$I_{st}(\mathbf{s}_i) = \min\left(p_{st} + O_{in,st}(\mathbf{s}_i) + r_{st}(\mathbf{s}_i), K_{\mathrm{u},st}(\mathbf{s}_i)\right), \text{ with}$$

$$I_{st}(\mathbf{s}_i) = \frac{I(\mathbf{s}_i)}{m_{K_\mathrm{u}(\mathbf{s})}},\ p_{st} = \frac{p}{m_{K_\mathrm{u}(\mathbf{s})}},\ O_{in,st}(\mathbf{s}_i) = \frac{O_{in}(\mathbf{s}_i)}{m_{K_\mathrm{u}(\mathbf{s})}}, \qquad (9)$$

$$r_{st} = \frac{r}{m_{K_\mathrm{u}(\mathbf{s})}},\ K_{\mathrm{u,st}}(\mathbf{s}_i) = \frac{K_\mathrm{u}(\mathbf{s}_i)}{m_{K_\mathrm{u}(\mathbf{s})}}$$

$$O_{in,st}(\mathbf{s}_i) = \sum_{n=1..m} O_{out,st}(\mathbf{s}_{i,n}), \qquad (10)$$

$$O_{out,st}(\mathbf{s}_i) = O_{in,st}(\mathbf{s}_i) + p_{st} + r_{st}(\mathbf{s}_i) - I_{st}(\mathbf{s}_i) \qquad (11)$$

where 'min(x,y)' assigns the minimum value of x and y. The subscripts '*st*'denote values standardized to values valid for a unit mean saturated conductivity, which is needed since the transfer function will be defined for standardized values, too. These are (see also Figure 6.1C): p_{st}, rainfall (h^{-1}); $I_{st}(\mathbf{s}_i)$, actual infiltration of the subunit (h^{-1}); $O_{out,st}(\mathbf{s}_n)$, outflow to a directly neighboring subunit (h^{-1}); $O_{in,st}(\mathbf{s}_i)$, total inflow from neighboring units 1..m to the subunit (h^{-1}); $K_{\mathrm{u},st}(\mathbf{s}_i)$, saturated conductivity of the subunit (h^{-1}); $r_{st}(\mathbf{s}_i)$, runon over the edge of the unit (h^{-1}).

Subunits at the edge of the unit may receive runon ($r_{st}(\mathbf{s}_i)$) from neighboring units while subunits in the center of the unit do not receive runon from neighboring units (Figure 6.1C). The amount of runon received by all subunits at the edge of the unit equals the total inflow from neighboring units in the rainfall-runoff model. It is assumed that each subunit at the edge of the unit receives an amount of this inflow proportional to its supplying catchment area outside the unit (Figure 6.1B):

$$\left.\begin{array}{ll} r_{st}(\mathbf{s}_i) = \dfrac{\mathrm{B}_i}{\sum_{i\in edge}\mathrm{B}_i} \cdot \dfrac{|\mathrm{u}|}{|\mathrm{u}_d|} q_{st}, & \text{for } \mathrm{B}_i > 0 \\ r_{st}(\mathbf{s}_i) = 0 & \text{for } \mathrm{B}_i = 0 \end{array}\right\}, \qquad (12)$$

with: B_i, catchment area outside the unit of a subunit i at the edge of the unit (m^2); $\sum_{i\in edge}\mathrm{B}_i$, catchment area of subunit i outside the unit, cumulative over all subunits at the edge of the unit (m^2); $|u|$, area of a subunit (m^2); $|u_d|$, area of a unit in the dynamic spatial rainfall-runoff model (m^2); $q_{st} = q / m_k$, with q (mm/h, over the unit), the inflow from neighboring units in the dynamic spatial rainfall-runoff model. The catchment area B_i is

the catchment area outside the unit, in neighboring units (Figure 6.1B). B_i is calculated using the flow pattern between subunits in neighboring units derived from the elevation model with random noise in neighboring units following the procedure also applied for the unit itself (section 6.4.1). B_i is the area represented by subunits on one or more upstream paths over the local drain direction network between the subunits, starting at these subunits at the edge of the unit receiving inflow from a neighboring unit.

The standardized effective saturated conductivity $K_{e,st}$ (h^{-1}) of the unit is:

$$K_{e,st} = \frac{\sum_{i=1..n} I_{st}(\mathbf{s}_i)}{n}. \quad (13)$$

with n, the number of subunits in the unit. And it is clear that the effective saturated conductivity K_e (mm/h) of the unit is

$$K_e = K_{e,st} m_{K_u(\mathbf{s})} \quad (14)$$

Also, the model calculates A_i (-), which is the area inside the unit which is downstream of one or more subunits at the edge of the unit, divided by $|u_d|$. This downstream area is calculated using the flow pattern between subunits.

The model G is represented by equations 4-14. It derives K_e and A_i from: (1) the process model given by equations 9-14, (2) the distribution of the input random field $K_u(\mathbf{s}_i)$ and the random flow pattern, (3) a rain intensity p, which is homogeneous within the unit, and the inflow from neighboring units q. The Monte Carlo simulation approach solves G in two steps (*Hammersley and Handscomb*, 1979; *Heuvelink*, 1998). Step 1. Repeat M times: a) Generate a realization of the random field $K_u(\mathbf{s}_i)$ and the flow pattern between the subunits $u(\mathbf{s}_i)$, b) With this realization of $K_u(\mathbf{s}_i)$ and the flow pattern, run the model with a fixed amount of rainfall p and inflow q, and store the model outcome (realization) of K_e. Step 2. The M model realizations represent K_e, and can be used to calculate parameters describing the distribution of K_e. The same holds for A_i.

6.5 The transfer function

The form of the transfer function (g) needs to be derived from infiltration and runoff processes occurring in a model unit as described by the stochastic model G. A fraction a (-) of the total area of the unit receives inflow from upstream units, which corresponds in the stochastic model G to the area represented by all subunits downstream of subunits at the edge of a unit receiving inflow from neighbouring units. The effective saturated conductivity of the unit can be written as the weighted sum of the effective saturated conductivity ($k_{e,ch}$) in the compartment receiving inflow, represented by the fraction a, and the effective saturated conductivity ($k_{e,nc}$) in the compartment receiving net rain only, represented by 1-a:

$$k_e = a \cdot k_{e,ch} + (1-a) \cdot k_{e,nc} \quad (15)$$

Under the invalid assumption of 1) a complete distribution of all inflow and net rain within each compartment, and 2) absence of spatial variation in saturated conductivity within the two compartments, k_e is:

$$k_e = a \cdot \min(p + \frac{q}{a}, m_{k,ch}) + (1-a) \cdot \min(p, m_{k,nc}), \quad (16)$$

with, $m_{k,ch}$ and $m_{k,nc}$ the saturated conductivity within each of the two compartments. In most cases, the assumption can be made that $m_{k,ch}$ and $m_{k,nc}$ (mm/h) are the same, as is done in the stochastic model G. Then, after some manipulation, equation 16 becomes:

$$k_e = \left\{ \min[p_{st},1] + \min[\, q_{st}, a(1 - \min[p_{st},1])\,] \right\} \cdot m_{K_u(\mathbf{s})},$$

with: (17)

$$p_{st} = \frac{p}{m_{K_u(\mathbf{s})}}, \text{ and } q_{st} = \frac{q}{m_{K_u(\mathbf{s})}},$$

and $m_{K_u(\mathbf{s})}$, the mean saturated conductivity of the unit. Figure 6.3 illustrates the shape of equation 17, using $k_{e,st} = k_e / m_{K_u(\mathbf{s})}$.

With spatial variation of inflow and saturated conductivity, k_e is lower than represented in equation 17, and the 'min' function is replaced by a function h, with additional input parameters b_p and b_q:

$$\begin{aligned} &k_e(p_{st}, q_{st}) = \mathrm{g}(p_{st}, q_{st}) \cdot m_{K_u(\mathbf{s})}, \text{ with} \\ &\mathrm{g}(p_{st}, q_{st}) = \left\{ \mathrm{h}[p_{st},1,b_p] + \mathrm{h}[q_{st}, a(1 - \mathrm{h}[p_{st},1,b_p]), b_q] \right\} \end{aligned} \quad (18)$$

with b_p, b_q, parameters defining the shape of the function in the p_{st} and q_{st} direction respectively.

The next question is to find a form for g, which is equivalent to finding a form for h. The form of h needs to follow from the properties of the stochastic model G defined by eq. 3-14. Some main properties of h can be derived from G as follows. Let Δ be a random variable representing the total standardized runon to a subunit:

$$\Delta = p_{st} + O_{in,st}(\mathbf{s}_i) + r_{st}(\mathbf{s}_i) \quad (19)$$

It is shown in Appendix 6.1, lemma 1, that for I the following limit is true (L. Booth, Utrecht University, personal communication):

$$\lim_{\delta \to 0} \mathrm{E}\left(\frac{I(\Delta)}{\Delta} \middle| \Delta \le \delta \right) = 1 \quad (20)$$

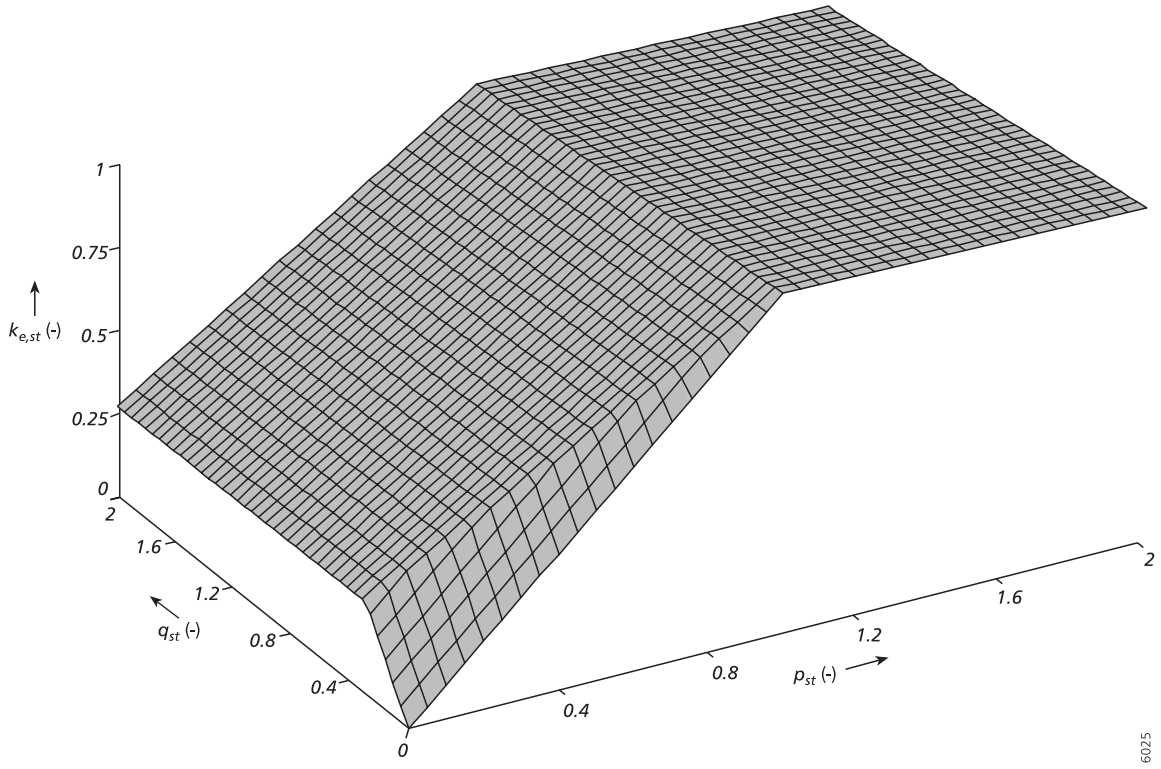


Figure 6.3. Relation between q_{st}, p_{st}, and $k_{e,st}$ for equation 17, a = 0.28. Note that $k_{e,st} = k_e / m_{K_u(\mathbf{s})}$.

with δ, a variable; and $I(\Delta)$, the actual infiltration of a subunit as a function of Δ, see appendix 6.1 for a further explanation. From eq. 20 it follows that g should have the following properties:

$$\left.\frac{\partial g[p_{st}, q_{st}]}{\partial p_{st}}\right|_{(p_{st},q_{st})=(0,0)} = 1, \quad \left.\frac{\partial g[p_{st}, q_{st}]}{\partial q_{st}}\right|_{(p_{st},q_{st})=(0,0)} = 1 \qquad (21)$$

In addition, Appendix 6.1, lemma 2, shows that the following limit is true (L. Booth, Utrecht University, personal communication):

$$\lim_{\delta \to +\infty} \mathrm{E}\left(I(\Delta) \middle| \Delta \geq \delta\right) = \mathrm{E}\left(K_{\mathrm{u},st}(\mathbf{s}_i)\right) \qquad (22)$$

And from this it follows that g should have the following properties:

$$\lim_{p_{st} \to +\infty} g[p_{st}, q_{st}] = 1, \ \lim_{q_{st} \to +\infty} g[0, q_{st}] = a \qquad (23)$$

A function h resulting in the properties of g given in equation 21 and 23 is given by (Appendix 6.2, L. Booth, Utrecht University, personal communication):

$$h[x,d,b] = d\,\frac{1+b+\frac{x+bx}{d}-\sqrt{-4b(1+b)\frac{x}{d}+\left((1+b)\left(1+\frac{x}{d}\right)\right)^2}}{2b} \qquad (24)$$

Appendix 6.2 gives proofs for the properties of the function g given in equation 21 and 23. The surface in Figure 6.4 illustrates the shape of the transfer function, with an example using the same input parameter for a used in Figure 6.3, and rather low values for b_p and b_q. When b_p and b_q are increased the surface approaches the surface represented by equation 17 (Figure 6.3). In other words, b_p and b_q define how much 'lower' the surface is compared to the surface represented by equation 17, in the p_{st} and q_{st} directions respectively. Another example is given in Figure 6.6.

A sensitivity analysis shows that with a p_{st} value greater than q_{st}, typical for the first part of a rainstorm, $k_{e,st}$ calculated by g(p_{st},q_{st}) is more sensitive to changes of b_p than to changes in other input parameters (Figure 6.5A), since infiltration in the area (a) with runon from upstream cells is very low as a result of the small value of q_{st}, resulting in a low sensitivity to changes in b_q. For a p_{st} value less than the q_{st} value, the opposite is true, and $k_{e,st}$ becomes also sensitive to changes in b_q and a (Figure 6.5B).

6.6 Parameterization of the transfer function

6.6.1 Fitting to outcomes of stochastic model for within-unit runoff-infiltration modeling

Parameter values of the transfer function g (equation 18) for each unit of catchment B are derived by fitting the transfer function to the outcome of the stochastic model G. For each unit, the following procedure is followed: 1) The stochastic model is run for 64 different combinations of values for p_{st} and q_{st} in the range of values actually occurring during a rainstorm, using M = 40 Monte Carlo samples per combination, resulting in 64·40 = 2560 realizations, 2) the parameter a in the transfer function is assigned the value E$\{A_i\}$, 3) the remaining parameters b_p and b_q are fitted to the 2560 realizations of K_e using least squares regression, constraining both parameters to an upper bound of $1{\cdot}10^4$. At the upper bound value, the difference between the shape of the transfer function and equation 17 is negligible, and a further increase does not result in a significant change in shape of the transfer function.

The procedure was followed with different combinations of inputs: 1) four different scenarios of runoff patterns: ploughing, sheet flow, wheel tracks perpendicular to the general flow direction and wheel tracks parallel to the general flow direction on the hillslope; 2) three different values of the semivariogram range parameter a_{var} = 1.25, 2.5, and 5 m; and 3) three different values of $\sigma_{Z(\mathbf{s})}$ = 0.48, 1.93, and 7.73. For each unit on catchment B, this resulted in 4 x 3 x 3 = 36 scenarios of the stochastic model with associated sets of parameter values a, b_p and b_q. Figure 6.4 and 6.6 give example fits of the parameters of g to the realizations of the stochastic model G, for two different units, while Figure 6.7 gives resulting maps for catchment B. These figures represent the results

for the scenario with flow with wheel tracks perpendicular to the general aspect of the field and input parameters a_{var} = 1.25, $\sigma_{Z(\mathbf{s})}$ = 7.73, which are the values derived from the field measurements. Note that these results will also be used for the runs with the dynamic spatial rainfall-runoff model.

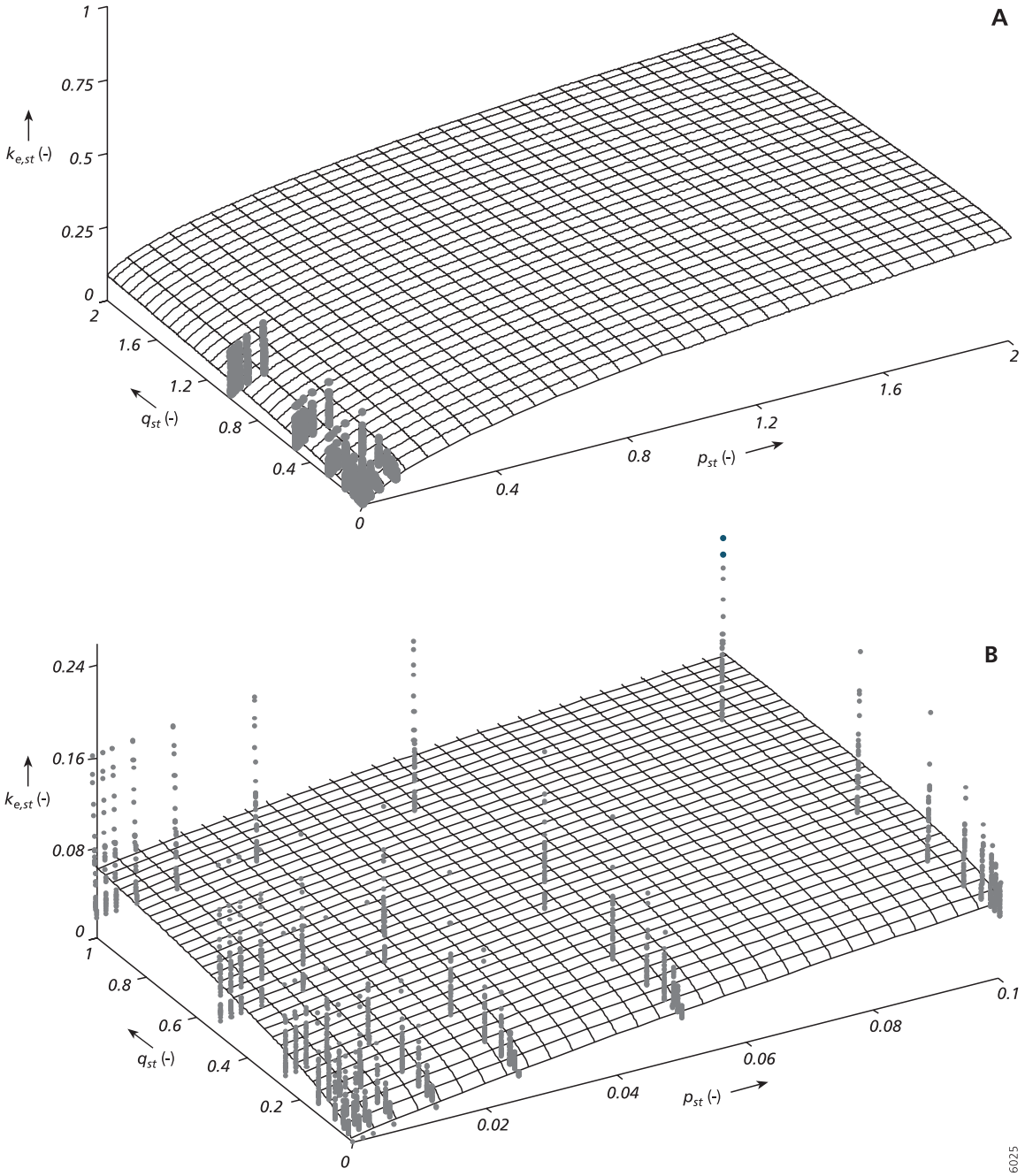


Figure 6.4. Surface: relation between q_{st}, p_{st}, and $k_{e,st}$ for the transfer function, a = 0.28 (which is the same as in Figure 6.3), b_p = -0.88, b_q = -0.98. Dots represent realizations of the Monte Carlo simulation to which b_p and b_q have been fitted. (A) axes corresponding to Figure 6.3; (B) cut off axes. Note that $k_{e,st} = k_e / m_{K_u(\mathbf{s})}$.

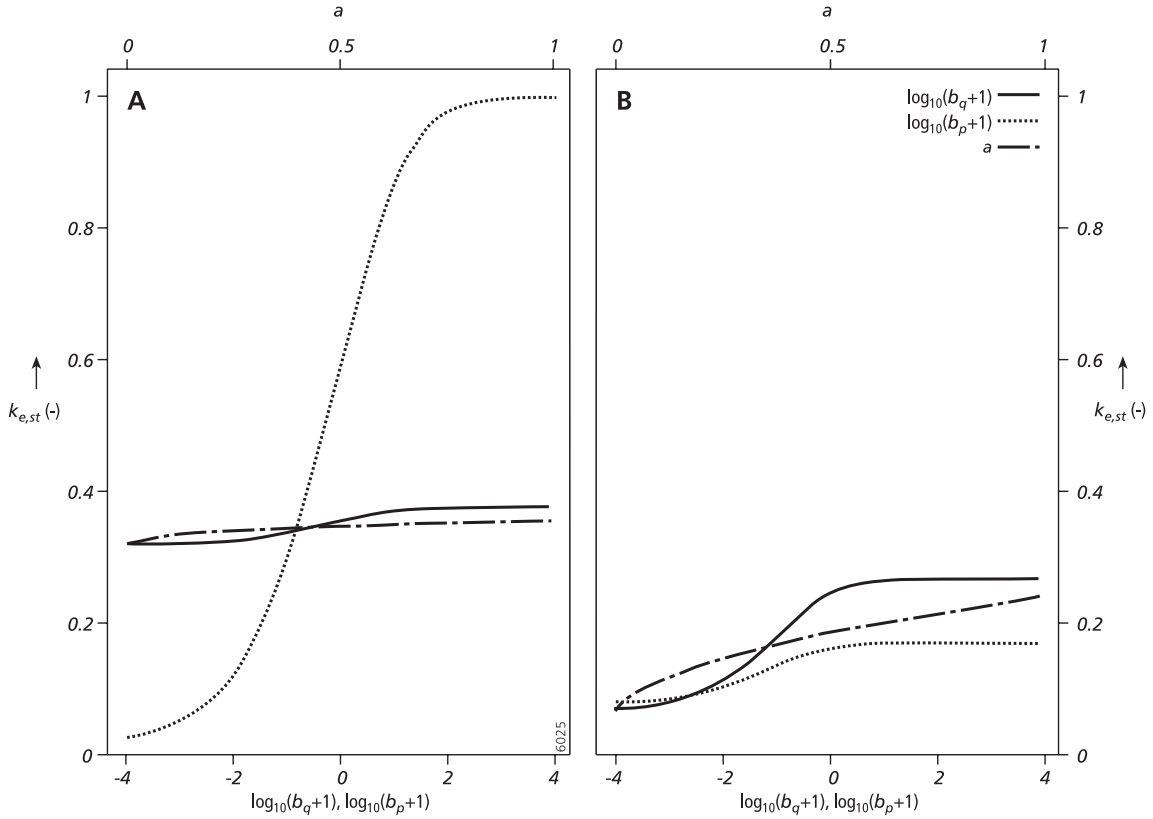


Figure 6.5. Sensitivity analysis with the transfer function. Each line represents the change in $k_{e,st}$ (i.e., $g(p_{st},q_{st})$) as a result of changing either a, $\log_{10}(b_p+1)$ or $\log_{10}(b_q+1)$, while keeping the other parameters constant at a standard value. For each parameter, this standard value is the average of the values on catchment B shown in Figure 6.7. These are a = 0.22, $\log_{10}(b_q+1)$ = -0.86, $\log_{10}(b_p+1)$ = -1.15. Upper horizontal axis, absolute values of a; lower horizontal axis, absolute values of $\log_{10}(b_p+1)$ and $\log_{10}(b_q+1)$. (A), p_{st} = 1.4, q_{st} = 0.1; (B), p_{st} = 0.1, q_{st} = 1.4.

6.6.2 Regression with derivatives of the digital elevation model

To parameterize larger catchments (e.g., catchment A), it is not feasible to run the stochastic model G for all model units. Instead, parameter values can be assigned using results of a regression analysis from catchment B. For each 10 x 10 m model unit used for rainfall-runoff modeling on catchment B, the following derivatives of the digital elevation model at that resolution are calculated: d_s, the topographical slope (m/m) at the unit, derived from a window of 3 by 3 units using the third-order finite difference method of *Horn* (1981), also applied by *Skidmore* (1989); d_c, the catchment area (m^2) of the unit, which is the area of the unit itself plus the units upstream over the local drain direction network derived with 8-point pour algorithm (*Burrough and McDonnell*, 1998); d_{pl}, topographical planform curvature, which is the curvature transverse to the slope, derived from a 3 by 3 window of units (*Zevenbergen and Thorne*, 1987); and d_{pr}, topographical profile curvature, which is the curvature in the direction of the slope, derived from

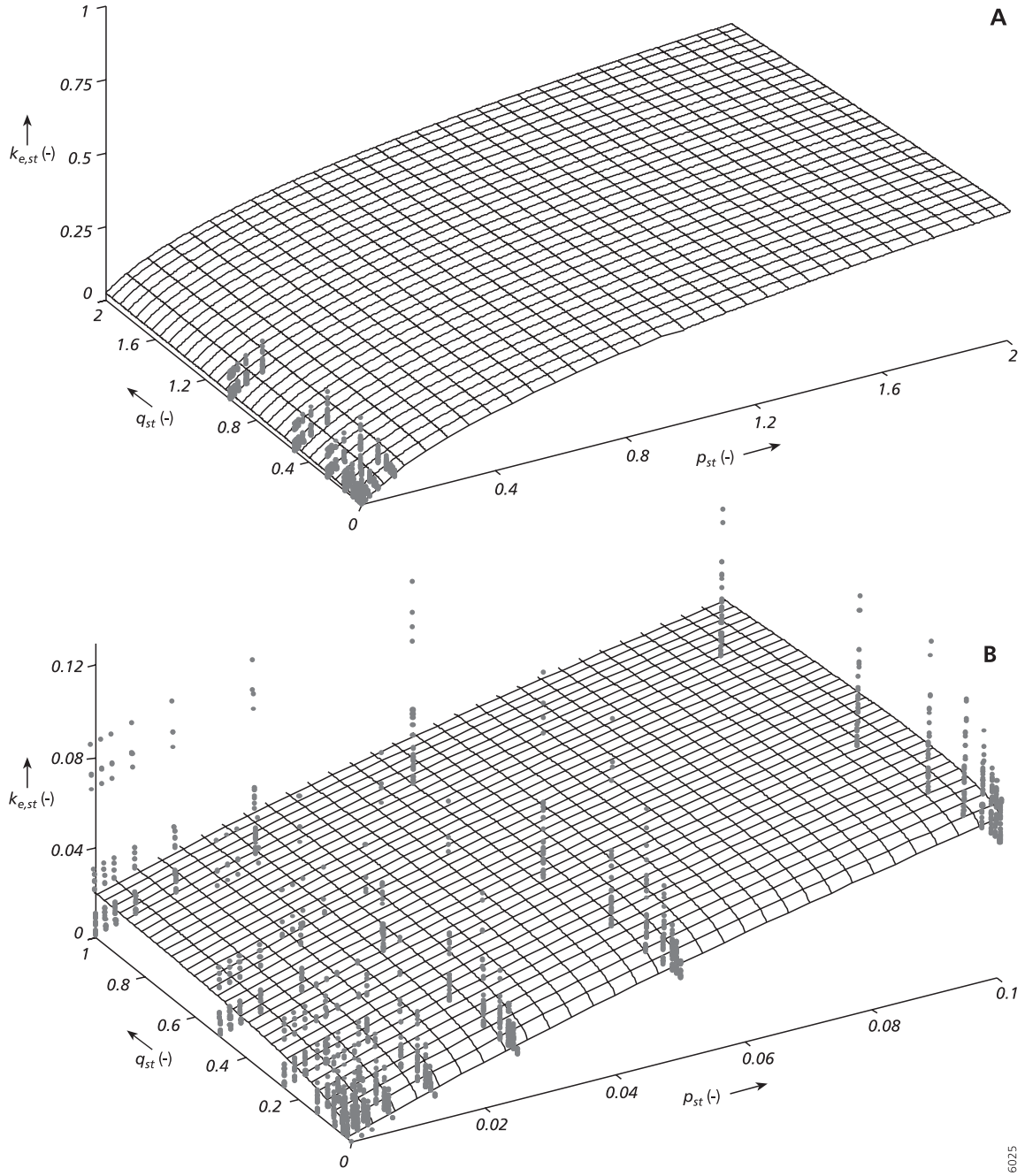


Figure 6.6. Surface: relation between q_{st}, p_{st}, and $k_{e,st}$ for the transfer function, a = 0.027, b_p = -0.78, b_q = -0.91. Dots represent realizations of the Monte Carlo simulation to which b_p and b_q have been fitted. (A) axes corresponding to Figure 6.3; (B) cutoff axes. Note that $k_{e,st} = k_e / m_{K_u(\mathbf{s})}$.

heights in a 3 by 3 window of units (*Zevenbergen and Thorne*, 1987). Both d_{pl} and d_{pr} have a positive value for a convex slope, and a negative value for a concave form of the slope. Finally, d_r is calculated, the topographical slope perpendicular to the ploughing direction divided by the slope parallel to the ploughing direction, calculated using the ploughing direction, the topographical slope d_s, and the direction of the steepest

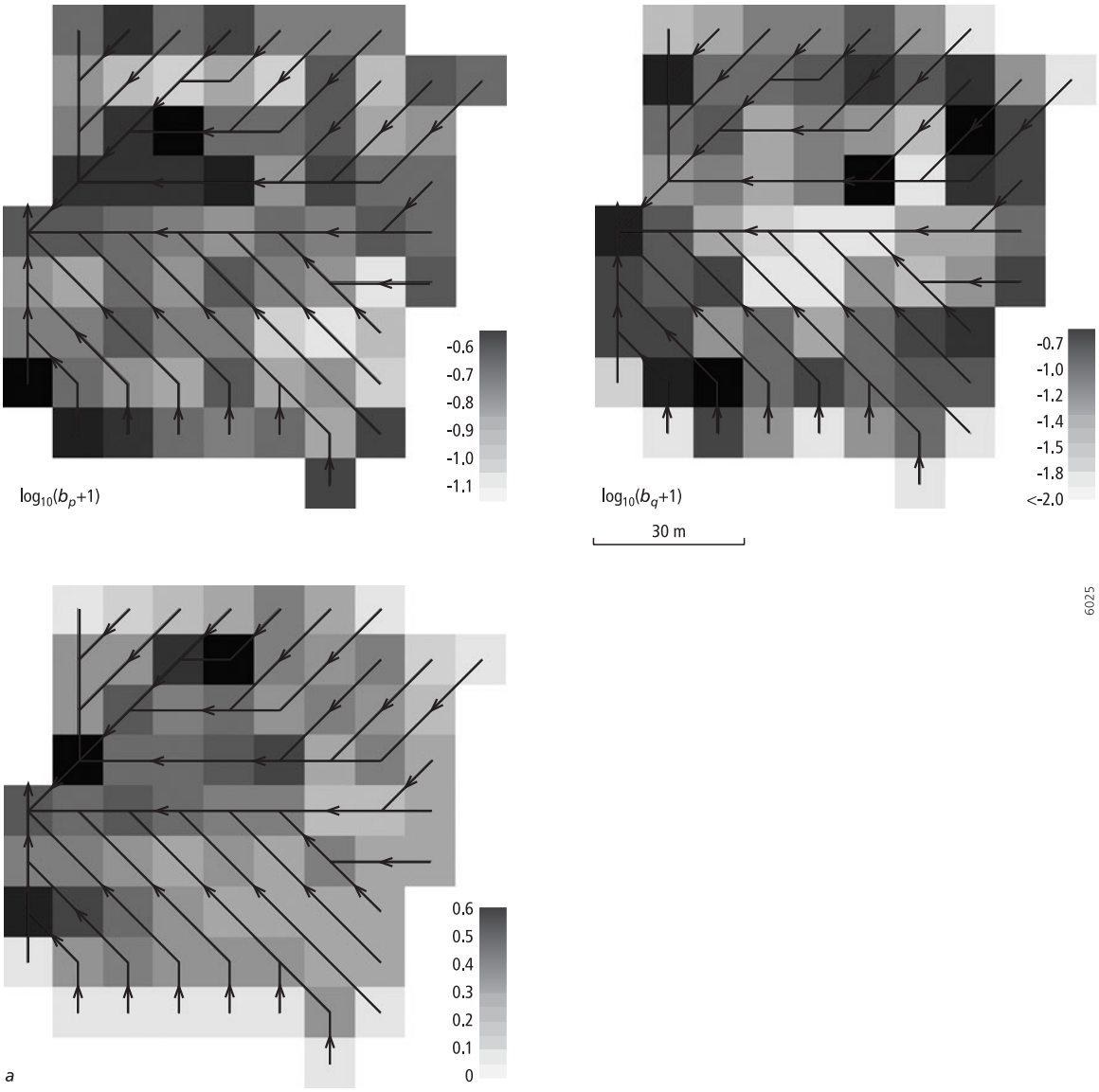


Figure 6.7. Maps with a, $\log_{10}(b_p+1)$ and $\log_{10}(b_q+1)$ values on catchment B, derived with the scenario with wheeltracks perpendicular to the general aspect of the field and a_{var} = 1.25, $\sigma_{Z(\mathbf{s})}$ = 7.73, which are the values determined by the field measurements.

topographical slope (i.e. aspect), calculated by the third-order finite difference method of *Horn* (1981). Values of d_{pr} are calculated for both the scenarios of the stochastic model with a ploughing direction parallel and perpendicular to the general aspect of the topography of catchment B.

Using the fitted model parameters resulting from the scenarios with the stochastic model, a, $\log_{10}(b_p)$ and $\log_{10}(b_q)$ were used as dependent variables in multiple linear regressions with d_s, $\log_{10}(d_c)$, d_{pl}, d_{pr}, $\log_{10}(d_r+0.001)$, $\sigma_{Z(\mathbf{s})}$ and a_{var} as independent variables. Log transformation was done for b_p, b_q, d_c and d_r, since log transformed values were closer to a normal distribution than untransformed values, based on visual

interpretation of histograms. For the multiple regressions, a backward elimination procedure was followed using the C_p statistic as a criterion for dropping variables in each regression (*Draper and Smith*, 1981). An inspection of the C_p plot for all possible regressions (*Draper and Smith*, 1981) confirmed for all regressions the choice of variables derived by the backward elimination procedure.

Although interpretation of the results in Table 6.2 is difficult, some conclusions can be drawn. The multiple correlation coefficients for the regressions are between 0.25 and 0.70, which means that the variation in the input parameters of the transfer function can partly be explained with derivatives of a digital elevation model at the resolution of the units and parameters describing the spatial probability distribution of $K_u(\mathbf{s})$. For the parameter a, the standardized regression coefficients (sometimes called beta weights) have the highest absolute value for d_c, the catchment area of a unit. The value of a is

Table 6.2. Coefficients for multiple linear regression with (log transformed) parameters of the transfer function as dependent variables and derivatives of the digital elevation model, $\sigma_{Z(\mathbf{s})}$, and a_{var} as independent variables (for description see text), catchment B, three scenarios: sheet flow, flow with microscale variation and flow with wheel tracks. Upper values are coefficients for non standardized independent and dependent variables, lower values are coefficients for independent and dependent variables standardized to one standard deviation (sometimes called beta weights). 'Int', intercept; r^2 is multiple correlation coefficient; n.s., non significant (dropped variables); - not used.

	Int	d_s	$\log_{10}(d_c)$	d_{pl}	d_{pr}	$\log_{10}(d_r+0.001)$	$\sigma_{Z(\mathbf{s})}$	a_{var}	r^2
Sheet									
a	-0.35	1.2	0.29	14	n.s.	n.s.	-	-	0.39
	-1.4	0.094	0.50	0.14	n.s	n.s	-	-	
$\log_{10}(b_p+1)$	1.25	-4.1	0.72	n.s.	n.s.	n.s.	-0.45	-0.14	0.59
	0.66	-0.045	0.17	n.s	n.s	n.s	-0.74	-0.12	
$\log_{10}(b_q+1)$	-0.025	3.74	-0.32	40	-266	n.s.	-0.13	-0.049	0.25
	-0.021	0.065	-0.12	0.083	-0.34	n.s	-0.34	-0.065	
Microscale									
a	0.0090	n.s.	0.036	1.7	n.s.	n.s.	-	-	0.38
	0.29	n.s	0.51	0.13	n.s	n.s	-	-	
$\log_{10}(b_p+1)$	1.0	-2.3	0.71	-27	37	n.s.	-0.48	-0.077	0.70
	0.55	-0.026	0.17	-0.037	0.031	n.s	-0.82	-0.066	
$\log_{10}(b_q+1)$	-0.55	1.4	n.s.	n.s.	-44	n.s.	-0.14	-0.058	0.61
	-0.96	0.052	n.s	n.s	-0.12	n.s	-0.75	-0.16	
Wheel tracks									
a	-0.24	1.6	0.14	6.7	n.s.	0.016	-	-	0.52
	-2.0	0.27	0.50	0.13	n.s	0.15	-	-	
$\log_{10}(b_p+1)$	1.5	n.s.	0.63	n.s.	33	0.13	-0.49	-0.15	0.65
	0.76	n.s	0.14	n.s	0.025	0.040	-0.79	-0.12	
$\log_{10}(b_q+1)$	-0.19	4.4	-0.23	49	-157	-0.079	-0.13	-0.050	0.29
	-0.20	0.093	-0.10	0.12	-0.24	-0.048	-0.43	-0.080	

mainly determined by the number of subunits at the edge of a unit receiving inflow from neighbouring cells. For units near the edge of the hillslope, the number of subunits receiving inflow is low compared to the other units since the units near the edge have a limited (or zero) number of upstream neighbor units. This causes a positive relation between a and d_c. The correlations between a and the planform curvature (d_{pl}) are always positive, which can be explained by a divergent flow pattern with positive planform curvature values and a convergent flow pattern with negative values. With convergent flow, a will be smaller than with divergent flow. The coefficients of the multiple regression for b_p and b_q are negative for $\sigma_{Z(\mathbf{s})}$ and a_{var}. This confirms the results described in chapter 5. In chapter 5, a decrease in effective saturated conductivity with an increase of these parameters was found, using a similar model. In addition, a number of derivatives of the elevation model are significant variables in the regressions for b_p and b_q.

Figure 6.8 illustrates the sensitivity of g(p_{st},q_{st}) (i.e. $k_{e,st}$) to changes of the independent variables in the multiple regression. For the derivatives of the digital

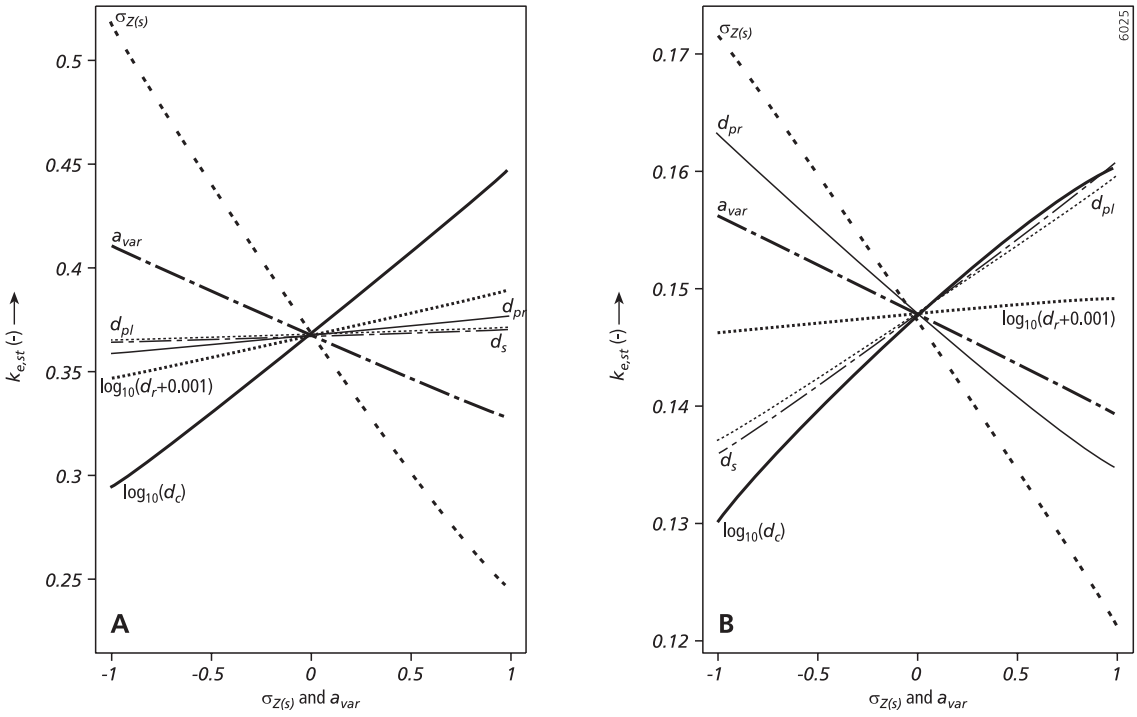


Figure 6.8. Sensitivity analysis with the multiple regression analysis for flow with wheel tracks as input to the transfer function. The value of $k_{e,st}$ (i.e., g(p_{st},q_{st})) is derived with the transfer function, by calculating its input parameters a, b_p, b_q with the multiple regression coefficients in Table 6.2, for flow with wheel tracks. Each line represents the change in $k_{e,st}$ (i.e., g(p_{st},q_{st})) as a result of changing one of the independent variables in the multiple regression. For d_s, $\log_{10}(d_c)$, d_{pl}, d_{pr}, and $\log_{10}(d_r+0.001)$), a value of 0 at the x-axis represents the average value, while a value of 1/-1 represents the average value plus/minus one standard deviation of the values on the hillslope. For $\sigma_{Z(\mathbf{s})}$, and a_{var}, a value 0 represents the measured mean value and 1/-1 represents this measured mean value plus or minus 1. (A), p_{st} = 1.4, q_{st} = 0.1; (B), p_{st} = 0.1, q_{st} = 1.4.

elevation model, it shows that $k_{e,st}$ is most sensitive to changes in d_c, while d_{pl} and d_s have an additional significant effect for the case of large amounts of runoff and low amounts of rain (Figure 6.8B). The magnitude of change in the sensitivity analysis for $\sigma_{Z(\mathbf{s})}$ and a_{var} was arbitrary chosen, but can be regarded as representative for the values found in the field. Changing $\sigma_{Z(\mathbf{s})}$ and a_{var} with this magnitude has an effect on $k_{e,st}$ which is approximately equally large as changes resulting from the sensitivity analysis with derivatives of the digital elevation model, as shown in Figure 6.8.

For application in the dynamic spatial rainfall-runoff model for catchment A, the attributes used as independent variables in the multiple regression and the coefficients in Table 6.2 are used to estimate $k_{e,st}$ for each unit of catchment A, using derivatives of the digital elevation model of catchment A at the resolution of 10 x 10 m. For d_c, only the upstream area of cells with flow over an arable field is included, excluding the upstream area connected by small ditches occurring in the catchment.

6.7 Application of the transfer function in a rainfall-runoff model

6.7.1 The dynamic spatial rainfall-runoff model

For each time step of 5 seconds, the net rainfall (p_n, m per time step) reaching the ground is:

$$p_n = p_r - c \cdot j \qquad (25)$$

with; p_r, the rain (m/timestep); c, percentage of the grid cell covered with vegetation (m^2/m^2); and j (m/timestep), the amount of water transported to the interception store, in meters per timestep for the area covered with vegetation. j is (*Merriam*, 1973):

$$j = s_m \left(e^{\left(\frac{-p_{c,t-1}}{s_m} \right)} - e^{\left(\frac{-p_{c,t}}{s_m} \right)} \right), \qquad (26)$$

with: s_m, maximum content of interception store (m); $p_{c,\,t-1}$, cumulative net rainfall since the start of the rainfall, at preceding time step (m); $p_{c,t}$, cumulative net rainfall (m) since start of the rainfall, at current time step. Since some units of catchment A contain ditches, surface water that is not kept in a constant potential surface storage is routed as sheet flow towards a ditches-network with channel flow, using the kinematic wave and the Manning equation (*Li et al.*, 1975; *Chow et al.*, 1988). The transfer function (g, eq. 18) parameterized for each unit is used to derive $k_{e,st}$ for the area with sheet flow, for each unit and each timestep, and note that $k_e = k_{e,st} \cdot m_{K_u(\mathbf{s})}$. For each t (s) in the model, a steady state situation of rainfall and runoff is assumed over a period (t-Δt,t), with Δt = 100 s, which is the estimated average travel time of water as sheet flow through a unit. For each timestep at t, the runon at t from sheet flow areas in upstream cells, excluding runon through ditches, is used for q_{st} in the transfer function, assuming that q at t represents the

average situation of surface flow over (t-Δt,t). For each timestep at t, the value of p_{st} is derived from the average value of p_n (eq. 25) over the period (t-Δt,t).

For each unit, and each time step, actual infiltration in the area with sheet flow is the minimum value of k_e (for that timestep and unit) and the amount of net rainfall (p_n) plus runon from the sheet flow area in upstream cells at the timestep. For the ditches occurring in the catchment, a constant infiltration capacity of 20 mm/h is used, which was found by *Van Dijck* (2000) in catchment A. The ditches-network with channel flow takes into account the infiltration of runon to the channel in a particular unit from the sheet flow area within the unit and inflow from upstream units containing a ditch. The model was implemented in the PCRaster spatio-temporal modeling language (*Wesseling et al.*, 1996).

6.7.2 Results

For each catchment, four scenarios were run (Table 6.3). The *transfer* scenario uses the transfer function with $m_{K_u(\mathbf{s})} = 1004$ mm/h, which is the measured mean of the saturated conductivity, while the *transfer, cali* scenario uses the transfer function with a value $m_{K_u(\mathbf{s})}$ found by calibration, minimizing the mean square error:

$$MSE = \frac{\sum_{e=1..n}\left(l_e^2\right)}{n} \qquad (27)$$

with n, the number of events measured for the catchment, where $l_e = \log_{10}(d_s+0.1)-\log_{10}(d_m+0.1)$, with d_s, the simulated cumulative discharge at the outflow point (m^3), and d_m the measured cumulative discharge (m^3). The *fixed* and *fixed, cali* scenarios do not use the transfer function. Instead, it is assumed that the saturated conductivity for each unit is equal to $m_{K_u(\mathbf{s})}$ for all timestep. The *fixed* scenario uses the measured value of $m_{K_u(\mathbf{s})} = 1004$ mm/h, while the *fixed, cali* uses a value of $m_{K_u(\mathbf{s})}$ found by calibration, minimizing *MSE*. For each of the scenarios, Table 6.3 gives *MSE* and the mean error (*ME*):

$$ME = \frac{\sum_{e=1..n}\left(l_e\right)}{n} \qquad (28)$$

From an analysis of the scenarios without calibration, for which the results are given in Table 6.3, Figure 6.9 and 6.10, one can conclude that the *transfer* scenario gives better results than the *fixed* scenario for both catchments, although also the *transfer* scenario significantly underestimates runoff. Also after calibration, the model with the transfer function (*transfer, cali* scenarios) gives better results than the model without the transfer function (*fixed, cali* scenarios), and it can be concluded that the model with the transfer function gives better results regarding cumulative discharge than the model without.

Table 6.3. Mean error (*ME*), mean squared error (*MSE*) and mean of saturated conductivity ($m_{K_u(\mathbf{s})}$) for simulations with the rainfall-runoff model . Different scenarios for catchment A and B are shown: *transfer*, runs with the transfer function without additional calibration; *fixed*, runs with a fixed value $m_{K_u(\mathbf{s})}$ of saturated conductivity, without using the transfer function; *transfer, cali*, runs with the transfer function with calibration of $m_{K_u(\mathbf{s})}$; *fixed, cali*, runs with a fixed value of $m_{K_u(\mathbf{s})}$ found by calibration, without using the transfer function.

Scenario	ME	MSE	$m_{K_u(\mathbf{s})}$ (mm/h)
Catchment A			
transfer	-1.9	5.1	1004
fixed	-3.6	13.3	1004
transfer, cali	-0.2	0.2	40
fixed, cali	-0.7	3.6	4.4
Catchment B			
transfer	-1.5	2.6	1004
fixed	-2.0	4.2	1004
transfer, cali	0.3	0.4	25
fixed, cali	0.4	0.8	1.7

Calibration of $m_{K_u(\mathbf{s})}$ results in a value $m_{K_u(\mathbf{s})}$ which is much lower than the measured value of $m_{K_u(\mathbf{s})}$, both with and without the transfer function. This is probably caused by an overestimation of $m_{K_u(\mathbf{s})}$ derived from the ring infiltrometers and rainfall simulation experiments, since the value of $m_{K_u(\mathbf{s})}$ = 1004 mm/h is high compared to saturated conductivity values generally measured on this soil type. Additional causes of this discrepancy could be measurement errors related to other parameters used in the models. For most scenarios, the timing and shape of the hydrographs corresponded rather well with measured hydrographs, for scenarios generating runoff. A further evaluation of the shape of the hydrograph is not done here, since parameters determining the shape of the hydrograph (e.g. Manning's roughness coefficient) were not calibrated.

6.8 Discussion

In chapter 5, a similar steady-state scaling method was tested, a similar rainfall-runoff model, and the same data set for simulating discharge from catchment B. In contrast to the approach followed here, the research described in chapter 5 used effective values of saturated conductivity representative for a support with the size of catchment B as a whole in the simulations with the rainfall-runoff model. It was concluded that the approach followed in that study suffered from an absence of steady-state conditions of rainfall and runoff on the hillslope as a whole. An improvement of the results was expected when the saturated conductivity is scaled to a support representing a smaller area, and a shorter time slice. The latter approach was tested in this research, and results are somewhat better than to those found in chapter 5. Further research with other data sets is necessary to evaluate both approaches regarding their usefulness in transient rainfall-runoff modeling. Testing the approaches with other data sets is also important because the

large values for the average and the skewness of the distribution of saturated conductivity ($K_u(\mathbf{s})$), found by ring infiltrometer measurements in this research area, is not representative for many other catchments.

The upscaling method described here suffers from some weaknesses that need to be resolved. The first weakness is related to the calculation used to distribute the inflow (q_{st}) to a unit over the subunits at the edge of the unit, resulting in an amount of runon (r_{st}) for

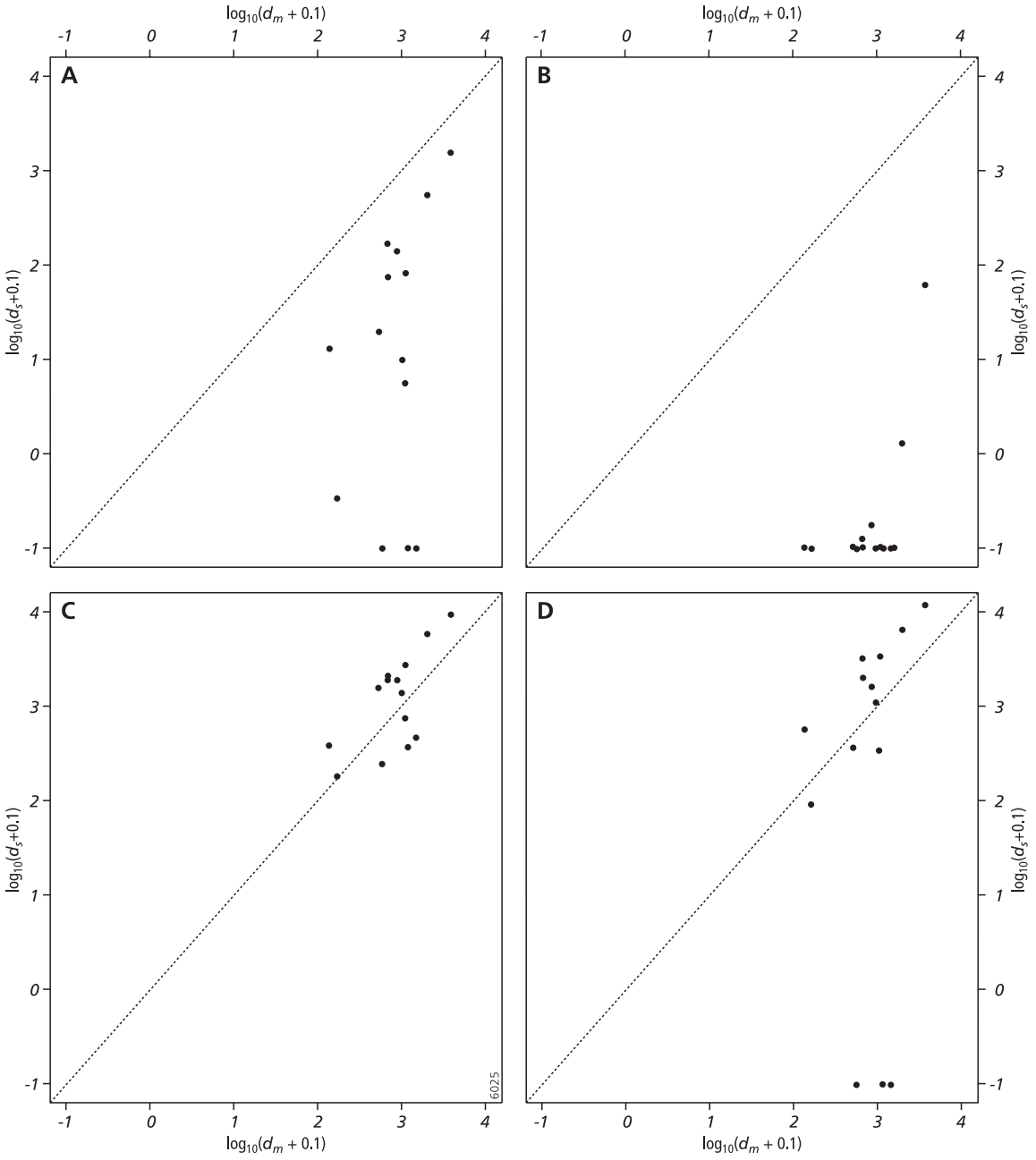


Figure 6.9. Catchment A, $\log_{10}$ of simulated cumulative discharge plus 0.1 (d_s+0.1, m^3) against measured cumulative discharge plus 0.1 (d_m+0.1, m^3). Measurement error in cumulative discharge, defined by $d_{m,min}$ and $d_{m,max}$, smaller than size of dots. (A) *transfer*, (B) *fixed*, (C) *transfer, cali*, (D) *fixed, cali* scenarios.

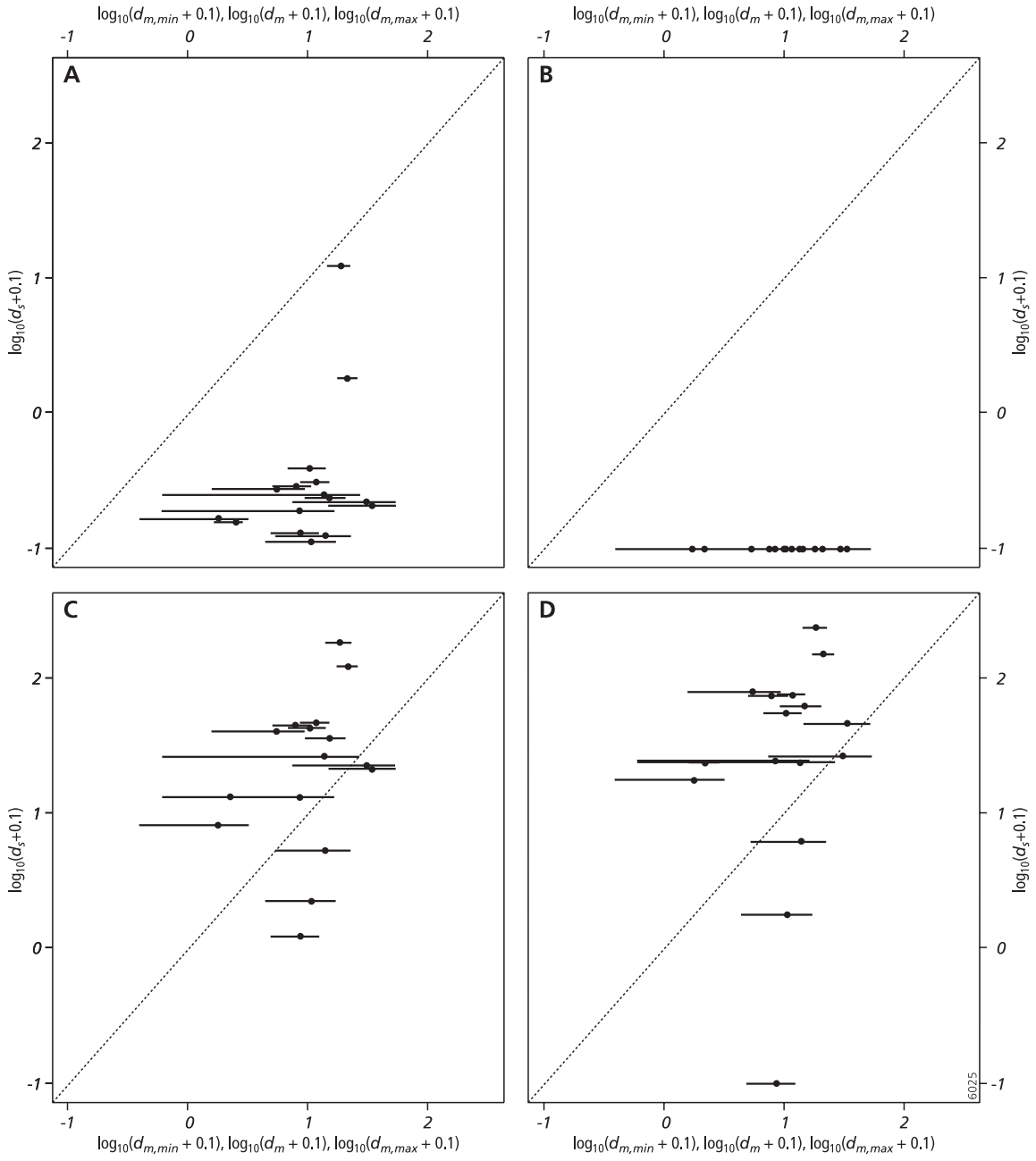


Figure 6.10. Catchment B, $\log_{10}$ of simulated cumulative discharge plus 0.1 (d_s+0.1, m^3) against measured cumulative discharge plus 0.1 ($d_{s,m}$+0.1, m^3), dots. Horizontal lines indicate measurement error, with endpoints of each line representing $d_{m,min}$ and $d_{m,max}$ of the measured discharge. (A) *transfer*, (B) *fixed*, (C) *transfer, cali*, (D) *fixed, cali* scenarios.

each subunit at the edge of the unit, see equation (12). For technical reasons, the assumption is made in equation (12) of an amount of runon (r_{st}) proportional to the catchment area (B_i) of the subunit. This results in an overestimation of runon for subunits with a large catchment area (B_i) because the interaction of rainfall and runoff in the catchment area is not taken into account in equation 12. Another weakness is the

assumption of a deterministic transfer function. At fixed values of q_{st} and p_{st}, the results in Figure 6.4 and 6.6 show that each Monte Carlo sample results in a different value of $k_{e,st}$. This spread is not represented by the transfer function, which models one value for each combination of p_{st} and q_{st}. This weakness could be overcome by using stochastic variables for a, b_p, and b_q in the transfer function, with parameter distributions fitted to the results of the Monte Carlo simulation. This approach would need a stochastic rainfall-runoff model, too. This weakness related to the transfer function becomes less significant when smaller values of $\sigma_{Z(\mathbf{s})}$ and a_{var} (a_{var}, with respect to the size of the unit) are used, since the spread in the values of $k_{e,st}$ becomes smaller for these cases (results not shown).

Another weakness of the approach is that the infiltration capacity of a unit at the start of a rainstorm is greater than $k_{e,st}$, as a result of the suction force, which is not represented in the upscaling method. For the case study described here, the suction force is assumed to have a negligible effect on infiltration, since the process of infiltration is mainly macro pore flow, as shown by the high average value of $k_{e,st}$ derived from ring infiltrometer experiments. For other catchments, this may not be the case.

6.9 Conclusions

The research showed that it is possible to develop a transfer function that calculates effective values of saturated conductivity for model units of approximately 100 m^2 from local scale (approximately 0.04 m^2) values of saturated conductivity. The form of the transfer function fits results of a stochastic steady-state runoff-infiltration model calculating the same scale transfer of saturated conductivity.

The parameters in the transfer function correlate with parameters describing the spatial probability distribution of the saturated conductivity within a model unit, derivatives of the digital elevation model at the scale of the model unit (e.g., slope, curvature), and the pattern of surface runoff in the unit. Using this correlation, it is possible to parameterize the transfer function for each model unit of a dynamic spatial rainfall-runoff model.

With this parameterization, it was shown that the application of the transfer function in a dynamic spatial rainfall-runoff model resulted in differences between simulated and measured cumulative runoff which were smaller than these found with simulations using the same rainfall-runoff model without the transfer function. This was the case both for the catchment consisting of one hillslope, and for the larger catchment, consisting of several hillslopes. Although the transfer function is derived from the stochastic steady-state runoff-infiltration model, the reasonably good results of its application in a dynamic spatial rainfall-runoff model mentioned above indicates that it can be applied in transient simulations.

Acknowledgements
Lorna Booth (Department of Mathematics, Utrecht University, the Netherlands) is acknowledged for a number of significant inputs to this research. Many thanks to Simone van Dijck (currently at Alterra, Wageningen, the Netherlands) for providing the data set,

and comments to an earlier version of the manuscript. Peter Burrough, Marc Bierkens (Utrecht Centre for Environment and Landscape Dynamics, Utrecht University), and Willem van Deursen (Carthago Consultancy, the Netherlands) are thanked for their constructive comments to an earlier version of the manuscript.

6.10 References

Aitchison, J. & J.A.C. Brown (1957), The lognormal distribution, with special reference to its uses in economics. Cambridge: Cambridge University Press.

Beven, K. (1989), Changing ideas in hydrology - the case of physically-based models, Journal of Hydrology 105, pp. 157-172.

Bierkens, M.F.P., P.A. Finke & P. de Willigen (2000), Upscaling and downscaling methods for environmental research. Dordrecht: Kluwer.

Binley, A. & K. Beven (1989), A physically based model of heterogeneous hillslopes: 2. Effective Hydraulic Conductivities. Water Resources Research 25, pp. 1227-1233.

Blöschl, G., R.B. Grayson & M. Sivapalan (1995), On the representative elementary area (REA) concept and its utility for distributed rainfall-runoff modelling, Hydrological Processes 9, pp. 313-330.

Blöschl, G. & M. Sivapalan (1995), Scale issues in hydrological modelling: a review. Hydrological Processes 9, pp. 251-290.

Burrough, P.A. & R.A. McDonnell (1998), Principles of Geographical Information Systems, Oxford: Oxford University Press.

Buttle, J.M. & D.A. House (1997), Spatial variability of saturated hydraulic conductivity in shallow macroporous soils in a forested basin. Journal of Hydrology 203, pp. 127-142.

Chow, V.T., D.R. Maidment & L.W. Mays (1988), Applied Hydrology. New York: McGraw-Hill.

Corradini, C., R. Morbidelli & F. Melone (1998), On the interaction between infiltration and Hortonian runoff. Journal of Hydrology 204, pp. 52-67.

Draper, N.R. & H. Smith (1981), Applied regression analysis. New York: Wiley.

Grant, S.A., J.D. Jabro, D.D. Fritton & D.E. Baker (1991), A stochastic model of infiltration which simulates “macropore” soil water flow. Water Resources Research 27, pp. 1439-1446.

Hammersley, J.M. & D.C. Handscomb (1979), Monte Carlo Methods. London: Chapman & Hall.

Harms, T.E. & D.S. Chanasyk (2000), Plot and small-watershed scale runoff from two reclaimed surface-mined watersheds in Alberta. Hydrological Processes 14, pp. 1327-1339.

Hendrayanto, K.K. & T. Mizuyama (2000), Scaling hydraulic properties of forest soils, Hydrological Processes 14, pp. 521-538.

Heuvelink, G.B.M. (1998), Error Propagation in Environmental Modelling with GIS. London: Taylor & Francis.

Horn, B.K.P. (1981), Hill shading and the reflectance map. Proceedings IEEE 69, pp.14-47.

Kamphorst, E.C., V. Jetten, J. Guérif, J. Pitkänen, B.V. Iversen, J.T. Douglas & A. Paz (2000), Predictiong Depressional Storage from Soil Surface Roughness. Soil Science Society of America, Journal, 64, pp. 1749-1758.

Li, R.-M., D.B. Simons & M.A. Stevens (1975), Nonlinear kinematic wave approximation for water routing. Water Resources Research 11, pp. 245-252.

Mallants, D., B.P. Mohanty, A. Vervoort & J. Feyen (1997), Spatial analysis of saturated hydraulic conductivity in a soil with macropores. Soil Technology 10, pp. 115-131.

Merriam, R.A. (1973), Fog drip from artificial leaves in a fog wind tunnel. Water Resources Research 9, pp. 1591-1598.

Merz, B. & E.J. Plate (1997), An analysis of the effects of spatial variability of soil and soil moisture on runoff. Water Resources Research 33, pp. 2909-2922.
Pebesma, E.J. & G.B.M. Heuvelink (1999), Latin hypercube sampling of Gaussian random fields. Technometrics 41, pp. 303-312.
Ragab, R. & J.D. Cooper (1993), Variability of unsaturated zone water transport parameters: implications for hydrological modelling. 2. Predicted vs. in situ measurements and evaluation of methods. Journal of Hydrology 148, pp. 133-147.
Russo, D. (1992), Upscaling of Hydraulic conductivity in Partially Saturated Heterogeneous Formation. Water Resources Research 28, pp. 397-409.
Russo, D. & E. Bresler (1981), Soil Hydraulic Properties as Stochastic Processes: I. An Analysis of Field Spatial Variability. Soil Science Society of America Journal 45, pp. 682-687.
Skidmore, A.K. (1989), A comparison of techniques for calculating gradient and aspect from a gridded digital elevation model. International Journal of Geographical Information Systems 3, pp. 323-334.
Smith, R.E. & R.H.B. Hebbert (1979), A Monte Carlo analysis of the hydrologic effects of spatial variability of infiltration. Water Resources Research 15, pp. 419-429.
Van Dijck, S.J.E. (2000), Effects of agricultural land use on surface runoff and erosion in a Mediterranean area. Utrecht: Koninklijk Nederlands Aardrijkskundig Genootschap/Faculteit Ruimtelijke Wetenschappen, Universiteit Utrecht. (Ph.D. thesis, Utrecht University)
Wesseling, C.G., D. Karssenberg, W.P.A. van Deursen & P.A. Burrough (1996), Integrating dynamic environmental models in GIS: the development of a Dynamic Modelling language. Transactions in GIS 1, pp. 40-48.
Woolhiser, D.A., R.E. Smith & J.-V. Giraldez (1996), Effects of spatial variability of saturated hydraulic conductivity on Hortonian overland flow. Water Resources Research 32, pp. 671-678.
Zevenbergen, L.W. & C.R. Thorne (1987), Quantitative analysis of land surface topography. Earth Surface Processes and Landforms 12, pp. 47-56.

Appendix 6.1

We consider here a particular subunit *i*. Let Δ be the standardised total runon to this subunit:

$$\Delta = p_{st} + O_{in,st}(\mathbf{s}_i) + r_{st}(\mathbf{s}_i)$$

Let $I(\Delta)$ be the standardised infiltration for this particular subunit, as a function of Δ, which is a certain standardised amount of total runon to the particular subunit. Let δ be a variable.

Proofs for the properties of the stochastic model

Lemma 1: $\lim_{\delta \to 0} \mathrm{E}\left(\frac{I(\Delta)}{\Delta} \middle| \Delta \le \delta \right) = 1$

Proof: note that $I(\Delta) \le \Delta$. Therefore:

$$\mathrm{E}\left(\frac{I(\Delta)}{\Delta} \middle| \Delta \le \delta \right) \le 1 \qquad (1)$$

Then observe:

$$\mathrm{E}\left(\frac{I(\Delta)}{\Delta} \middle| \Delta \le \delta \right) \ge \mathrm{E}\left(\frac{I(\Delta)}{\Delta} \middle| \Delta \le \delta \cap \Delta \le K_{\mathrm{u},st}(\mathbf{s}_i) \right) P\left(\Delta \le K_{\mathrm{u},st}(\mathbf{s}_i) \middle| \Delta \le \delta \right)$$

If $\Delta \le K_{\mathrm{u},st}(\mathbf{s}_i)$ then $I(\Delta) \le \Delta$, so

$$\mathrm{E}\left(\frac{I(\Delta)}{\Delta} \middle| \Delta \le \delta \right) \ge 1 \cdot P\left(\Delta \le K_{\mathrm{u},st}(\mathbf{s}_i) \middle| \Delta \le \delta \right) \ge P\left(K_{\mathrm{u},st}(\mathbf{s}_i) \ge \delta \right)$$

$\lim_{\delta \to 0} P\left(K_{\mathrm{u},st}(\mathbf{s}_i) \ge \delta \right) = 1$, and combined with eq. (1), Lemma 1 follows.

Lemma 2: $\lim_{\delta \to +\infty} \mathrm{E}\left(I(\Delta) \middle| \Delta \ge \delta \right) = \mathrm{E}\left(K_{\mathrm{u},st}(\mathbf{s}_i) \right)$

$$\begin{aligned} \mathrm{E}\left(I(\Delta) \middle| \Delta \ge \delta \right) &\ge \mathrm{E}\left(I(\Delta) \middle| \Delta \ge \delta \cap K_{\mathrm{u},st}(\mathbf{s}_i) < \delta \right) P\left(K_{\mathrm{u},st}(\mathbf{s}_i) < \delta \right) \\ &= \mathrm{E}\left(K_{\mathrm{u},st}(\mathbf{s}_i) \middle| \Delta \ge \delta \cap K_{\mathrm{u},st}(\mathbf{s}_i) < \delta \right) P\left(K_{\mathrm{u},st}(\mathbf{s}_i) < \delta \right) \\ &= \mathrm{E}\left(K_{\mathrm{u},st}(\mathbf{s}_i) \middle| K_{\mathrm{u},st}(\mathbf{s}_i) < \delta \right) P\left(K_{\mathrm{u},st}(\mathbf{s}_i) < \delta \right) \qquad (2) \end{aligned}$$

Observe that

$$\lim_{\delta\to\infty} \mathrm{E}\left(K_{\mathrm{u},st}(\mathbf{s}_i)\middle|K_{\mathrm{u},st}(\mathbf{s}_i)<\delta\right)=\mathrm{E}\left(K_{\mathrm{u},st}(\mathbf{s}_i)\right) \tag{3}$$

and that

$$\lim_{\delta\to\infty} \mathrm{P}\left(K_{\mathrm{u},st}(\mathbf{s}_i)<\delta\right)=1 \tag{4}$$

Combining equations 2, 3 and 4 we see

$$\lim_{\delta\to+\infty} \mathrm{E}\left(I(\Delta)\middle|\Delta\geq\delta\right)\geq \mathrm{E}\left(K_{\mathrm{u},st}(\mathbf{s}_i)\right) \tag{5}$$

By the fact that $I(\Delta)\leq K_{\mathrm{u},st}(\mathbf{s}_i)$ and thus

$$\mathrm{E}\left(I(\Delta)\middle|\Delta\geq\delta\right)\leq \mathrm{E}\left(K_{\mathrm{u},st}(\mathbf{s}_i)\middle|\Delta\geq\delta\right)= \mathrm{E}\left(K_{\mathrm{u},st}(\mathbf{s}_i)\right) \tag{6}$$

Lemma 2 follows by combining equations 5 and 6.

Appendix 6.2

Explanation of the form of the transfer function

Notice that

$$\mathrm{g}(p_{st},0)=\mathrm{h}\left[p_{st},1,b_p\right] \quad \text{and} \quad \mathrm{g}(0,q_{st})=\mathrm{h}\left[q_{st},a,b_q\right] \tag{7}$$

Lemma 1: $\left.\frac{\partial \mathrm{g}[p_{st},q_{st}]}{\partial p_{st}}\right|_{(p_{st},q_{st})=(0,0)}=1,\ \left.\frac{\partial \mathrm{g}[p_{st},q_{st}]}{\partial q_{st}}\right|_{(p_{st},q_{st})=(0,0)}=1$

Notice that $\left.\frac{\partial h[x,d,b]}{\partial x}\right|_{x=0}=1$, combined with equation 7, this gives:

$$\left.\frac{\partial \mathrm{g}[p_{st},0]}{\partial p_{st}}\right|_{p_{st}=0}=1 \text{ and } \left.\frac{\partial \mathrm{g}[0,q_{st}]}{\partial q_{st}}\right|_{q_{st}=0}=1$$

Lemma 2: $\lim_{p_{st}\to+\infty} g[p_{st}, q_{st}] = 1$, $\lim_{q_{st}\to+\infty} g[0, q_{st}] = a$

Notice that $\lim_{x\to+\infty} h[x, d, b] = d$ combined with equation 7, this gives

$\lim_{p_{st}\to+\infty} g[p_{st}, q_{st}] = 1$, $\lim_{q_{st}\to+\infty} g[0, q_{st}] = a$

PART III:

INVERSE MODELLING

7 CONDITIONING A PROCESS BASED MODEL OF SEDIMENTARY ARCHITECTURE TO WELL DATA

Derek Karssenberg (with Torbjörn Törnqvist and John S. Bridge)

Abstract: Prediction of sedimentary architecture for modeling of fluid flow in hydrocarbon reservoirs and aquifers is accomplished mainly using stochastic, structure-imitating models, because these can be conditioned to data from wells, seismic profiles, and outcrop analogs. This implies that modeled architecture fits all available observations. However, the sedimentary architecture simulated by such models is commonly unrealistic. Process-based (forward) models potentially provide more realistic prediction and understanding of sedimentary architecture, but these models are not widely used because conditioning to well, seismic or outcrop data is considered to be very difficult. We show here that conditioning of process-based models to well data is possible in principle, using a 3D alluvial-architecture model as an example. This model considers the formation of alluvial deposits as a predominantly deterministic process, with a single channel belt moving by avulsion over an aggrading floodplain. However, the initial floodplain topography is simulated by a random field, thus yielding different model output for each run. Monte Carlo simulation was used to produce model realizations that fit five hypothetical vertical wells within predetermined tolerance bands. Such simulation allows calculation of the probability of occurrence of channel-belt deposits for each 3D cell in the 3D block of sediments generated by the model, as well as the probability distributions of volumes of channel-belt deposits and connectedness ratios. Adding more conditioning wells increases the precision of model predictions. Application of this approach in practice will require a major effort, particularly in overcoming the anticipated large amounts of computing time.

Published as: Karssenberg, D., T. Törnqvist, J.S. Bridge (2001), Conditioning a process-based model of sedimentary architecture to well data. Journal of Sedimentary Research 71, pp. 868-879. Reproduced with permission.

7.1 Introduction

Determination of the volume and quality of hydrocarbon reservoirs and aquifers, and development of fluid production and management strategies, requires understanding of the geometry, orientation, proportion, and spatial distribution (i.e., architecture) of the various sediment types present. Information derived from geophysical profiles, cores, well logs, and well-test data is rarely sufficient to provide comprehensive 3D description and understanding of hydrocarbon reservoirs or aquifers. Recourse must normally be made to outcrop analogs and depositional models. A common approach is to use outcrop analogs to provide supplementary data on sedimentary architecture, and to use stochastic models conditioned by subsurface data to distribute the architectural elements in 3D

space (reviews by Bryant and Flint 1993; Koltermann and Gorelick 1996; North 1996; Anderson 1997).

Modeling approaches include structure-imitating methods and process-based methods. Structure-imitating models do not simulate processes of deposition. These models directly simulate the sedimentary architecture and generally include stochastic components. Methods used are indicator geostatistics (e.g., Journel 1983; Bierkens and Weerts 1994), simulated annealing (e.g., Deutsch and Cockerham 1994), Markov chains (e.g., Doveton 1994, Carle et al. 1998), and probabilistic rules defining the geometry and location of stratigraphic units, known as Boolean object models (e.g., Budding et al. 1992; Deutsch and Wang 1996; Hirst et al. 1993; Holden et al. 1998). An advantage of these models is that they are conditioned to observational data. However, adequate input parameters are difficult to obtain for simulating realistic depositional architectures (e.g., Tyler at al. 1994; Deutsch and Wang 1996; Holden et al. 1998).

Unlike the structure-imitating models mentioned above, three-dimensional process-based models, sometimes referred to as process-imitating models, simulate the sedimentary processes acting to produce a deposit (Koltermann and Gorelick 1996; Anderson 1997). Process-based models can be deterministic and/or stochastic, and empirical and/or theoretical. Examples of such models include random-walk sedimentation models of braided rivers (Webb 1994), models based on the fundamental equations of fluid flow and sediment transport (e.g., Bridge 1977, 1992; Tetzlaff and Harbaugh 1989; Stam 1996; Gross and Small, 1998), and avulsion-related alluvial-architecture models (e.g., Bridge and Leeder 1979; Mackey and Bridge 1995; Heller and Paola 1996). Process-based models are forward models in the sense that they predict the nature of deposits given a set of initial starting parameters. It is not known *a priori* what the deposits will look like. Advantages of three-dimensional process-based models are that they can help provide genetic interpretations of deposits and can predict more realistic sedimentary architecture than structure-imitating (stochastic) models. A perceived disadvantage of three dimensional process-based models, however, is that it is difficult or impossible to make the simulated deposits fit (or be conditioned to) observational data in sufficient detail in three dimensions (Clemetsen et al. 1990; North 1996; Koltermann and Gorelick 1996; Anderson 1997). Therefore, process-based models have had limited application in quantitative simulation of the architecture of hydrocarbon reservoirs or aquifers, although recent studies have shown that two-dimensional sequence stratigraphic models with simple process equations can be conditioned to well data (Bornholdt et al. 1999; Cross and Lessenger 1999). But such two-dimensional models have limited interpretive value in hydrocarbon reservoir characterization and geohydrology.

The advantages of structure-imitating and process based methods could be retained by combining them. One way of doing this would be the use of a range of outputs from process-based models to provide input for stochastic models. However, the view that three dimensional process-based models cannot be conditioned to observational data can be challenged.

It is demonstrated here that conditioning of three-dimensional process-based models to well data using an essentially trial-and-error approach is possible in principle, using an alluvial-architecture model as an example. The model, based on that of Mackey and

Bridge (1995), considers the formation of alluvial deposits as a single channel belt moves by avulsion over an aggrading floodplain. The model is partly stochastic in the sense that it has a stochastic random field (initial floodplain topography) as input yielding a different model outcome (realization) for each run. Monte Carlo simulation is used to calculate the probability of occurrence of channel-belt deposits in each voxel (3D cell) of the 3D block of sediments generated by the model, as well as the probability distributions of volumes of channel-belt deposits and channel-belt connectedness ratios.

7.2 Model Concepts

In this exploratory study, we use a slightly simplified version of the 3D alluvial-architecture model of Mackey and Bridge (1995). It was simplified to save computer time for conditioning the model to the wells. We developed the new model in the spatio-temporal modeling language that runs inside the PCRaster environmental modeling system (Wesseling et al. 1996). (PCRaster information and demonstration software including links to gstat available at http://www.geog.uu.nl/pcraster.) Stochastic simulations are performed with Gstat (Pebesma and Wesseling 1998). The model starts by calculating the initial floodplain topography and the initial geometry and location of the channel belt. Then, the model calculates for each time step representing the period between avulsions: (1) channel-belt and overbank aggradation (thickness, age and type of deposited sediment, i.e., either channel-belt or overbank deposit) and new surface topography; and (2) channel-belt avulsion location and new channel-belt location (including erosion of previous deposits by channel-belt incision).

7.2.1 Initial Floodplain Topography

The rectangular modeling area has a down-valley length L (m) and width W (m). It is discretized by square raster cells with constant cell length C (m) and cell center coordinates x, y (m; origin at the bottom left corner of the modeling area). The initial floodplain surface has a constant down-valley slope plus uncorrelated random noise representing local variation in elevation. The initial floodplain topography is the only stochastic input to the model. For each cell, the surface elevation at the start of the model run is

$$E(x,y,0) = S \cdot y + e(x,y), \qquad (1)$$

with:

$E(x, y, 0)$	surface elevation (m) at point x,y at time $t = 0$ (yr),
S	down-valley slope (-),
y	distance of the cell center from the downstream end of the modeling area (m),
$e(x, y)$	stochastic field: spatially uncorrelated random noise with variance σ and zero mean.

7.2.2 Channel-Belt Geometry and Initial Location

Throughout the model run, a single channel belt is active on the floodplain. Unaggraded channel-belt dimensions are defined by the bankfull channel depth d (m) and channel-belt width w (m), both constant in time. The inflow location of the channel-belt center is at the upstream center cell (i.e., inflow cell) throughout a model run. The initial channel-belt location on the floodplain is determined by first defining the cells representing the downstream path of the channel belt center following the local direction of maximum floodplain slope, using the 8-point pour algorithm (Moore 1996; Burrough and McDonnell 1998). Each channel-belt center cell i leads to a channel-belt center cell that is one of the n = 1…8 adjacent cells following the maximum downstream slope (Figure 7.1A). Local closed depressions are removed using the algorithm of Van Deursen (1995). The initial channel belt consists of the cells with a distance less than $w/2$ normal to this centerline (Figure 7.1B).

7.2.3 Channel-Belt and Overbank Aggradation

The aggradation rate in the channel belt (a, m/yr) is constant in space and time. Overbank aggradation rate decreases with distance from the channel-belt edge. For each point x,y and time step t, the aggradation rate in the overbank area $A(x, y, t)$ (m/yr) is:

$$A(x,y,t) = ac + a(1-c)\, e^{-D(x,y,t) / b}, \qquad (2)$$

where:

a	channel-belt aggradation rate (m/yr),
c	theoretical aggradation rate at infinite distance from the channel belt, expressed as a fraction of a (-),
$D(x, y, t)$	distance to the channel-belt edge at time step t (m),
b	dimensionless aggradation exponent (-).

Mackey and Bridge (1995) did not include the c value in the equation for overbank aggradation rate. Equation (2) allows definition of aggradation rates that approach a value much greater than zero at large distances away from the channel belt that is not possible if c is omitted. If c is set to zero, the equation is equivalent to Mackey and Bridge's equation (1995), when D is converted to their dimensionless distance from the channel-belt edge.

For each time period between avulsions, the total thickness of aggraded sediment is stored by the computer model. This deposit is labeled with its age and sediment type (channel-belt deposit or overbank deposit). Erosion of formerly deposited strata occurs only as a result of incision of new channel belts. In the interest of simplicity, compaction of sediment is not included in this model (cf. Bridge and Leeder 1979; Mackey and Bridge 1995).

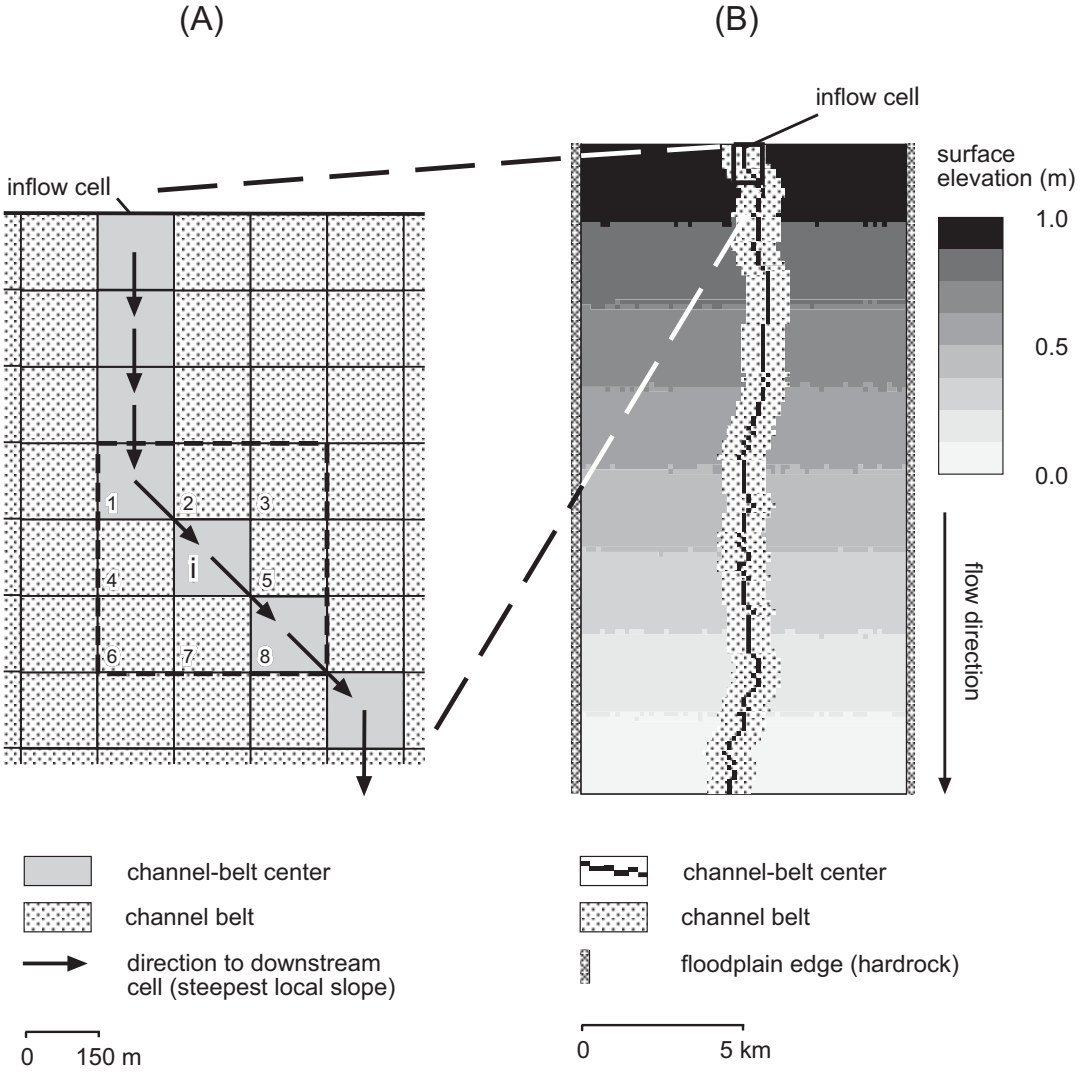


Figure 7.1. A) Calculation of channel-belt centerline that is the path in the direction of maximum floodplain slope. For cell *i*, the floodplain slope towards the 8 adjacent cells is calculated in order to determine the direction of maximum slope. B) Plan view of initial surface elevation $E(x, y, 0)$ (m), inflow cell, initial channel-belt centerline and initial channel belt.

7.2.4 Avulsion

Avulsion is the process whereby the channel belt shifts abruptly from one location to another on the floodplain. In the interest of model simplicity, avulsion occurs at a constant time interval T (yr). Concepts of calculation of the avulsion location correspond with Mackey and Bridge (1995), but the algorithms used are slightly different. The floodplain surface morphology at the edge of the channel belt determines the avulsion location (Figure 7.2). The direction of maximum floodplain slope at the edge of an unaggraded channel belt will be approximately parallel to the active channel belt, resulting in a low probability of avulsion. Because the aggradation rate of the channel

belt is greater than that of the overbank area, however, an alluvial ridge develops, and the direction of maximum floodplain slope may locally change from parallel to nearly perpendicular to the edge of an aggraded channel belt. This situation will lead to a high probability for an avulsion to occur (cf. Heller and Paola 1996; Slingerland and Smith 1998). This principle is represented in the model with the following algorithm. For each cell immediately neighboring the channel belt the avulsion angle α is calculated by $\alpha = | \beta - \gamma |$, with β, the azimuth of direction of maximum floodplain slope (algorithm of Horn 1981, also described in Burrough and McDonnell 1998), and γ, the azimuth of direction perpendicular to the channel-belt edge.

The avulsion location is the cell with the lowest avulsion angle (α) immediately adjacent to the channel belt (Figure 7.2). The lowest avulsion angle is equivalent to the highest value of Mackey and Bridge's (1995) slope ratio. The centerline of the new channel belt is determined as: (1) the set of cells that connect the avulsion cell with the center of the former channel belt (upstream path, A in Figure 7.3); (2) the center of the former channel belt upstream of this connecting set of cells (B in Figure 7.3); and (3) the downstream path from the avulsion cell over the locus of maximum floodplain slope (C in Figure 7.3) calculated with the 8-point pour algorithm also applied for the initial channel belt. The new channel belt consists of cells with a distance of less than *w*/2 normal to this centerline.

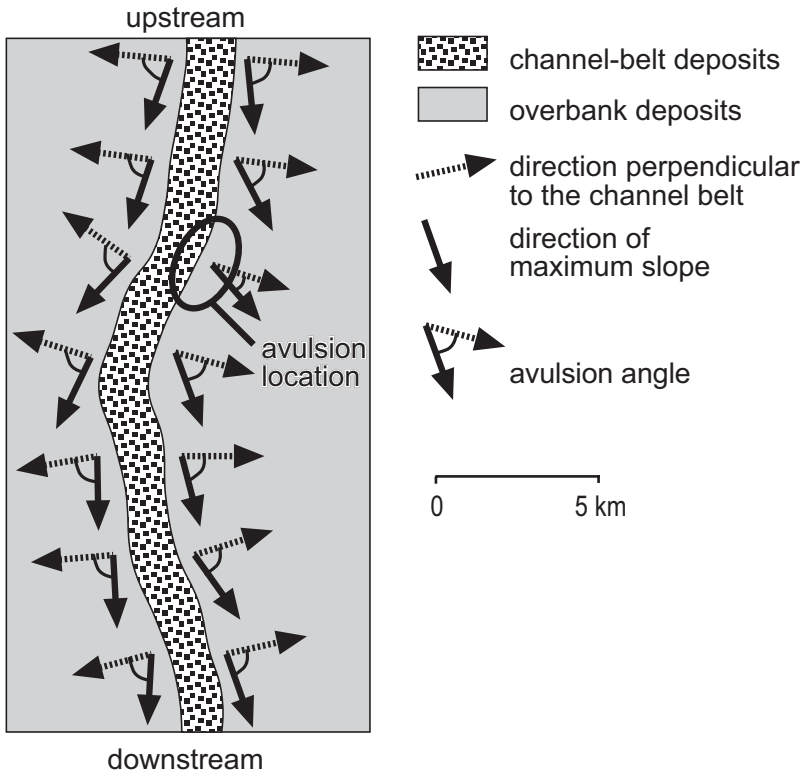


Figure 7.2. Determination of the location of the channel-belt avulsion.

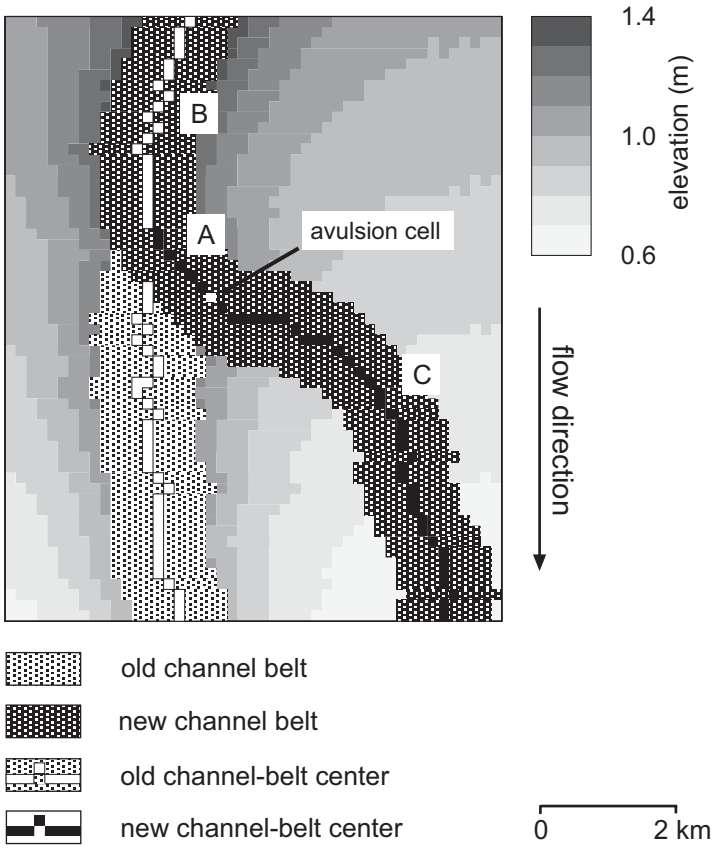


Figure 7.3. Calculation of new channel-belt center and new channel belt following an avulsion. Letters A, B, and C refer to explanation in text.

7.3 Conditioning To Well Data

Our model is stochastic only in the sense that the initial floodplain topography is represented in part by a random field, causing the first channel belt to be at a different location and elevation for each model run. As a result, the entire alluvial succession will be different for each run because the behavior of the model is determined by antecedent conditions. In fact, the probability that one model run will result in an outcome that agrees with well data is very low. The trial-and-error method for producing process-based model outcomes that agree with well data (well-conditioned output) is described below.

7.3.1 Stochastic Model with Output Conditioned to Well Data

The stochastic model resulting in an output conditioned to well data is

$$U(x,y,z) = g\big((E(x,y,0), d_1(x,y,z),...,d_m(x,y,z)\big) \quad (3)$$

and

$$f(U(x,y,z), \mathbf{w}) = 0 \quad (4)$$

where

$g(\cdot)$	the process-based alluvial-architecture model,
$E(x, y, 0)$	input random field: initial topography of the floodplain surface,
$d_1(x, y, z),...,d_m(x, y, z)$	deterministic input fields for the alluvial architecture model,
$U(x, y, z)$	output random field: well-conditioned alluvial architecture,
$f(\cdot)$	objective function for well-conditioned output,
w	well data.

The only input random field is $E(x, y, 0)$, the randomly varying initial floodplain elevation with random variation. All other parameters of the model are assigned deterministic values. The output random field is $U(x, y, z)$, which is the 3D block of sediment generated by the stochastic model at the end of the model run. In this block, each voxel has a probability of containing channel-belt deposits. The Boolean objective function f is a function of $U(x, y, z)$ and the well data **w**, giving an error of zero or one. If the error is zero, the outcome of the process-based model fits the well data within predetermined limits. If the error is one, the model outcome does not adequately fit the well data. The objective function is described in the next section.

The stochastic model derives the distributions (or parameters describing these) of the random field $U(x, y, z)$ from: (1) the distribution of the input random field $E(x, y, z)$ and the deterministic inputs $d_1(x, y, z),...,d_m(x, y, z)$; (2) the process-based model $g(\cdot)$; (3) the well data $\boldsymbol{w}$; and (4) the objective function $f(\cdot)$. The Monte Carlo simulation approach solves this in two steps (Hammersley and Handscomb, 1979; Heuvelink, 1998):

Step (1)
Repeat K times (lower case letters represent realizations):

a. Generate a realization of the initial floodplain elevation: input random field $e(x, y, z)$.
b. With this realization and deterministic input fields $d_1(x, y, z),...,d_m(x, y, z)$, run $g(\cdot)$ and compute the outcome $u(x, y, z)$ of the process-based model.
c. With the model outcome $u(x, y, z)$ and well data $\boldsymbol{w}$, calculate the error of the objective function f. If the error is 1, the outcome of the process-based model does not fit the well data and start again at (a); otherwise continue to (d).
d. Store the model outcome $u(x, y, z)$. N is the number of outcomes of the process-based model that resulted in well-conditioned output.

Step (2)
Compute sample statistics (e.g., mean, variance, skewness, channel-belt connectedness ratio, volume of channel belt deposits) from the N well-conditioned outcomes $u_{1..N}(x, y, z)$ of the process-based model.

7.3.2 Objective Function

The role of the objective function is to select the realizations of the process-based model that fit the well data within prescribed limits. These well-conditioned realizations are used for computing the sample statistics of the conditioned outcomes of the process-based model (e.g., channel-belt connectedness ratio). Running the model without the restrictions defined by the objective function results in a model output that does not fit the well data.

For each loop in the Monte Carlo simulation, the objective function is defined as

$$\mathrm{f}(u, w_1, \ldots, w_n) \begin{cases} = 0, & \text{if the generated alluvial succession } u \text{ fits all well logs } w_{1..n}, \\ = 1, & \text{if the generated alluvial succession } u \text{ does not fit one or more well logs } w_{1..n} \end{cases} \tag{5}$$

In the current model, we use vertical well logs with positions and thickness of the channel-belt and overbank deposits. For each well, the well log is compared with the alluvial succession generated by the process-based model at the vertical that contains the well. Figure 7.4 shows how the objective function is evaluated for each well. Figure 7.4A shows the well log. Because exact conditioning is extremely difficult, tolerance bands with vertical length w_t (Figure 7.4B) are defined at the boundary between the channel-belt deposits and overbank deposits, and at the top of the well log (top surface). The succession at the well location is said to fit the well data if the model output matches the strata in the well log within the tolerance bands, typically 0.1 - 1 m.

7.3.3 Reducing Computing Time with a Directive Function

In step (1) of the Monte Carlo simulation, only a small number of runs of the process-based model give output that matches well data, and a lot of computing time is needed because the process-based model has to be run thousands of times. For this reason, a directive function is included to decrease run times. This function is applied in step (1b) of the Monte Carlo simulation. The principle is that, while running the process-based model forward in time, for each $t = i$ the alluvial succession $u(t = i)$ is compared with the well data $w_1,\ldots,w_n$. If the alluvial succession for $t = i$ deviates from the well data in such a way that the model outcome will not be conditioned at the end of the model run, the run is interrupted and a new loop K is started by generating a new input random field (step 1a). Otherwise the process run is continued.

The directive function uses the property of the process-based model that channel-belt deposits cannot be replaced by overbank deposits, because erosion of the floodplain occurs only as a result of channel-belt incision. A run of the process-based model that

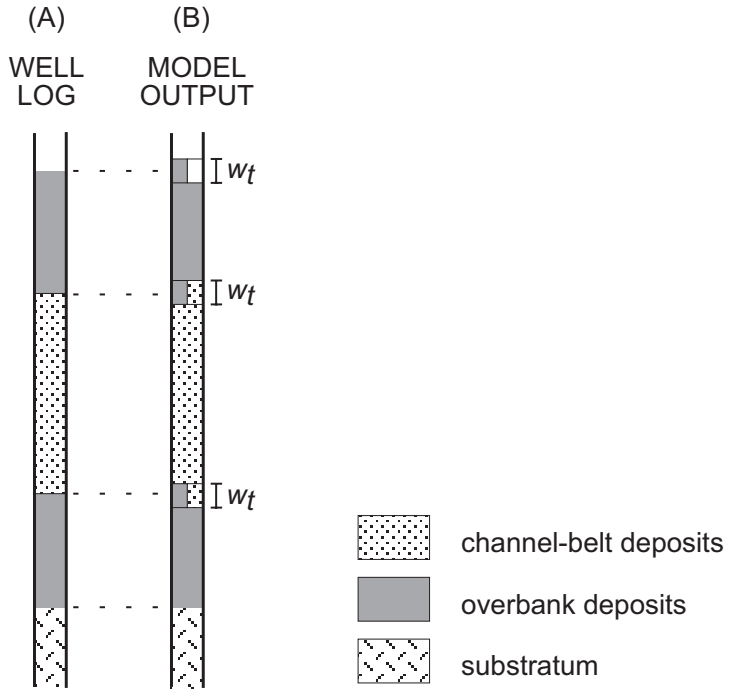


Figure 7.4. Evaluation of the objective function for each well. A) Well log and B) model output for conditioned result with tolerance bands w_t.

predicts channel-belt deposits in a well log at time step t where the well log contains overbank deposits cannot give a conditioned result, and the run of the process-based model is interrupted. As in the objective function, this comparison is done only for the well log outside the tolerance bands (Figure 7.5). Within the tolerance bands, the sediment type predicted by the model may deviate from the well log.

7.4 CASE STUDY

An example of an application of the approach, using model parameters given in Table 1, is now described. Figure 7.6 gives the hypothetical well data set used for conditioning. Model test runs demonstrated that a minimum value of 0.6 m is necessary for the width of the tolerance band. A further decrease in w_t resulted in unacceptably large computer run times to arrive at 50 realizations that fit the wells, needed for the stochastic model. Figure 7.7 gives the evolution in time of the floodplain for one run of the process-based model with the simulated deposits fitting the five wells. The initial floodplain at $t = 0$ shows a channel belt that is slightly curved as a result of the random noise included in the initial floodplain topography. The decrease in deposition rate with distance from the channel belt is represented in the topography at the end of a time step: an alluvial ridge is formed

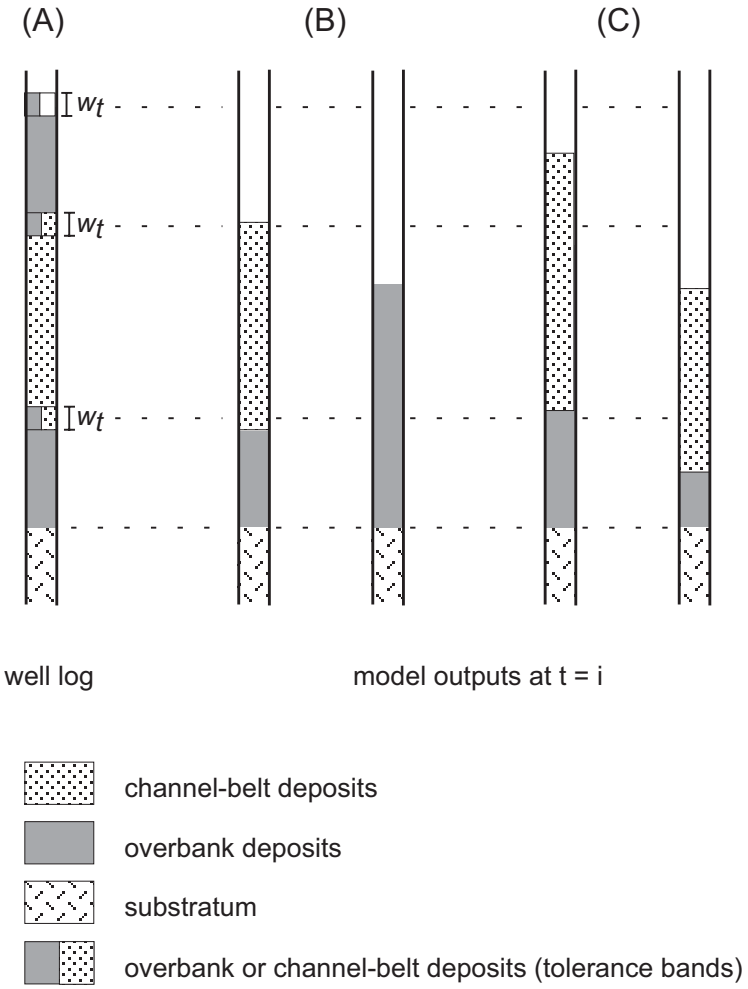


Figure 7.5. Evaluation of the directive function for each well. A) Well log with tolerance bands with width w_t. B) Model output at the well location for $t = i$ without directive function error: runs possibly resulting in conditioned model result at $t = end$. C) Model output at the well location for $t = i$ with directive function error: runs that will not result in conditioned model result at $t = end$.

at the location of the channel belt. As a result of this alluvial ridge, an avulsion occurs resulting in a new channel-belt location for the next time step. The maps show that the new channel belt for each time step follows the lowest topography downstream of the avulsion location. The simulated succession at the well sites matches the stratigraphy in the well logs within the tolerance bands (Figure 7.8). Figure 7.9 gives the 3D alluvial architecture resulting from the model run of Figure 7.7.

7.4.1 Effect of Number of Conditioning Wells on Model Output

In order to evaluate the effect of the number of conditioning wells on the model outcome, the stochastic model was run without conditioning and conditioned to different numbers

of wells. Figure 7.10 shows the stochastic outcomes of alluvial architecture for three scenarios, based on 50 realizations for each scenario (N = 50 in the Monte Carlo simulation procedure).

Without conditioning (Figure 7.10A), probabilities of greater than 0.8 for the occurrence of channel-belt deposits occur at the edges of the floodplain and close to the channel-belt inflow location. High probabilities at the edges of the floodplain are related to the decrease in deposition rate (hence floodplain elevation) with distance from the channel belt. An initial channel belt at the center of the floodplain results in the lowest deposition rate and elevation at the edge of the floodplain. Thus, subsequent channel belts tend to move towards these low areas (Figure 7.7). The area directly downstream of the inflow location has high probabilities because all channel belts originate in this zone. Between these high probability areas, values are typically 0.3 - 0.5 and show little spatial structure.

Addition of conditioning well data (Figure 7.10B, C) results in spatial heterogeneity occurring in the central part of the floodplain in addition to the pattern observed for the unconditioned run. Volumes with low and high probabilities for the occurrence of channel-belt deposits occur immediately adjacent to each other. The scenario conditioned to wells 1 and 2 (Figure 7.10B) results in an extensive volume with high probabilities of channel-belt occurrence upstream and downstream of well 1. Well 2 containing only overbank deposits results in low channel-belt probabilities near the well. The probability field for the scenario where all wells are used for conditioning is strongly determined by the five well logs (Figure 7.10C).

Table 7.1. Model parameter values for the case study.

Symbol	Value	Description
L	20100	down-valley floodplain length (m)
W	10050	cross-valley floodplain width (m)
C	150	cell length (m)
S	$5 \cdot 10^{-5}$	down-valley slope (-)
σ	$3 \cdot 10^{-3}$	variance of spatially uncorrelated random
d	10	bankfull channel depth (m)
w	1200	channel-belt width (m)
a	$2 \cdot 10^{-3}$	channel-belt aggradation rate (m/yr)
c	0.5	theoretical deposition rate at infinite distance from channel belt, fraction of a (-)
b	$1 \cdot 10^{3}$	overbank-aggradation exponent (-)
T	400	time interval between avulsions (yr)
w_t	0.6	width of tolerance bands (m)

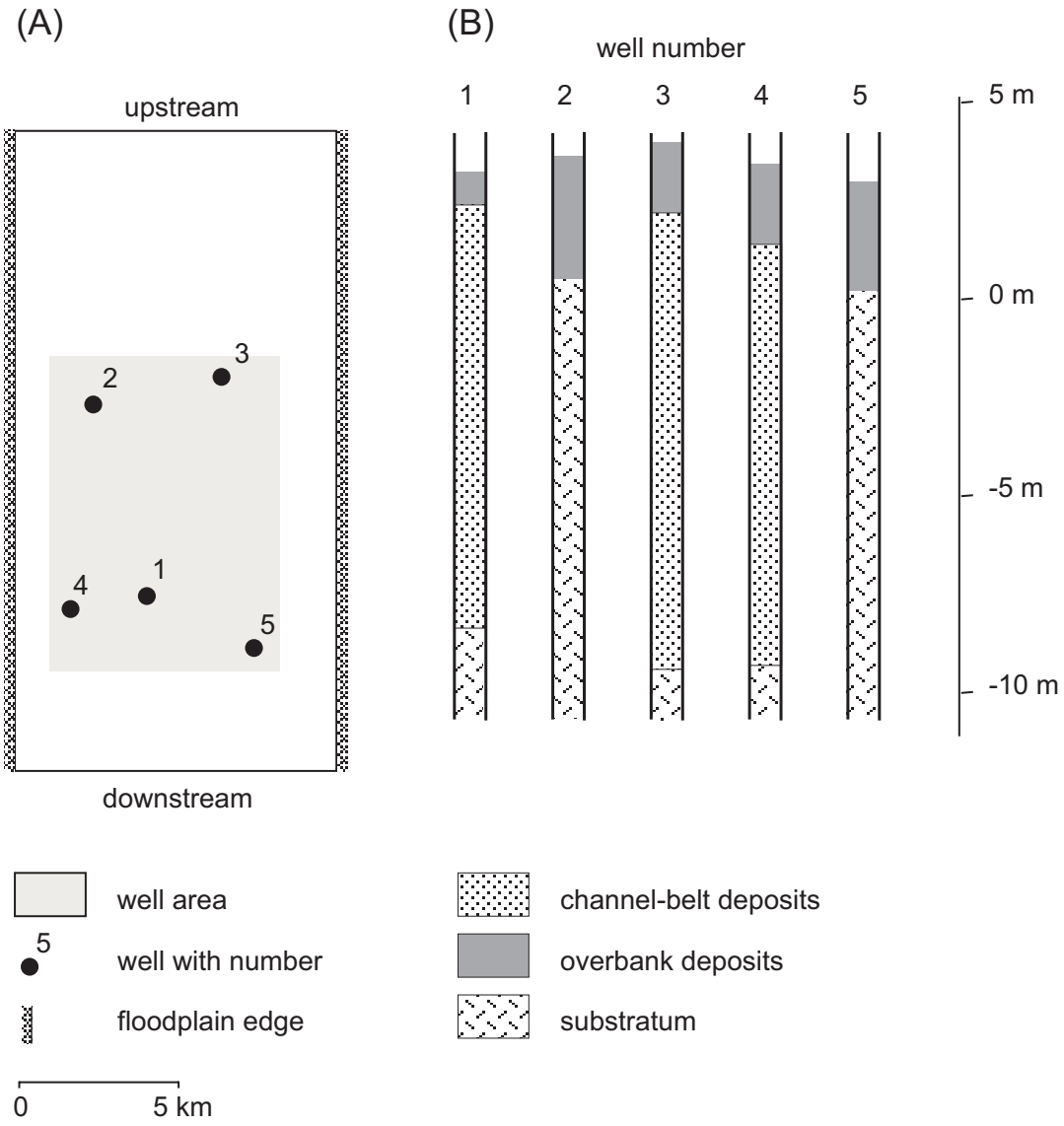


Figure 7.6. Case study well data set. A) Model area with the location of the wells and B) well logs. Well area is used for calculation of volume of channel-belt deposits and connectedness ratio. Origin of vertical axis corresponds to original topography (substratum) at downstream edge of the floodplain. Note that substantial amounts of substratum have been eroded at wells 1, 3, and 4.

7.4.2 Number of Well Logs and Estimation Precision

Total volume of channel belt deposits (m^3) and areal (2D) connectedness ratio were calculated within the area of the well data (Figure 7.6). Areal connectedness ratio is the sum for all channel belts of the total horizontal area of contact with another one, divided by the total horizontal area of all channel belts (Figure 7.11). Results for total volume of channel-belt deposits are reported here instead of net-to-gross ratio (channel-deposit

proportion), which can also be calculated. Net-to-gross ratios are not discussed because the thickness of the simulated deposit is too small to enable realistic comparison with real world successions.

Probability density distributions of total volume of channel-belt deposits and the areal connectedness ratios are unimodal (Figure 7.12). Average total volume of channel-belt deposits is between $4.0 \cdot 10^8$ and $4.8 \cdot 10^8$ m^3. Average connectedness ratios for the scenarios with different numbers of wells are between 0.4 and 0.5. There is no clear trend in these values with increasing number of wells.

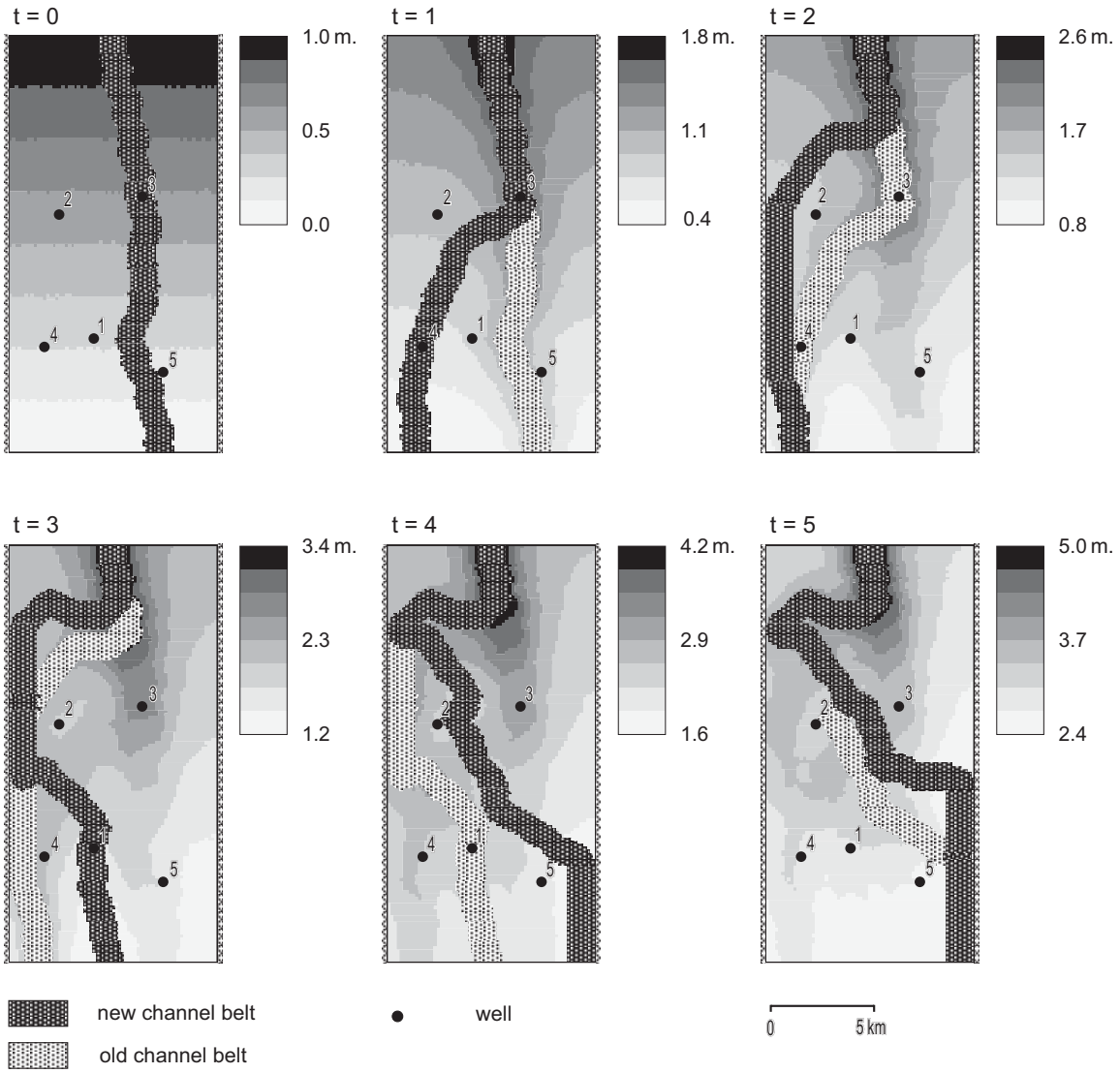


Figure 7.7. Evolution in time of conditioned model run, including initial situation (t = 0) and five successive time steps (each timestep represents 400 yr). Each map shows the position of the old and new channel belt, and surface topography (m) at the end of the time step, i.e., at the moment of the avulsion. The new channel belt corresponds with the old channel belt for the next time step.

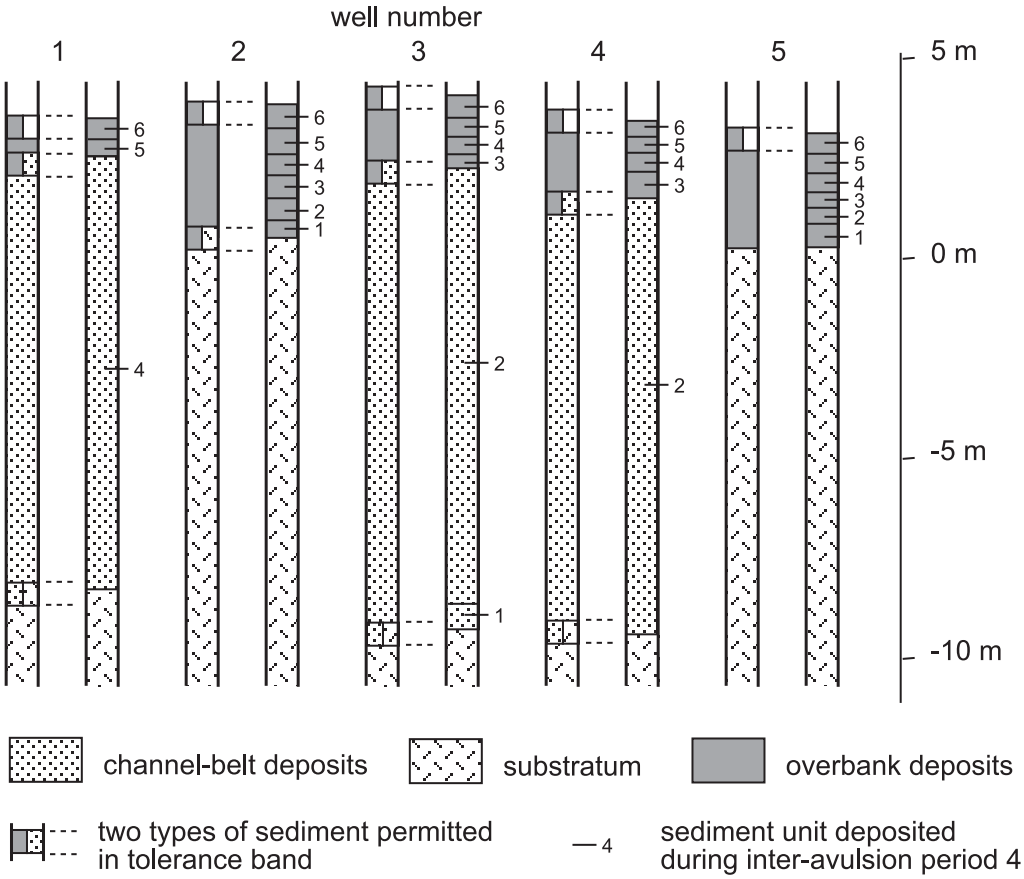


Figure 7.8. Conditioning to well data of the model run in Figure 7.7. For each well number the left column represents the well log with tolerance bands; the right column represents model output. Numbers to the right of the wells refer to interavulsion periods (i.e., time steps in Figure 7.7). Substratum is material below initial floodplain surface at $t = 0$.

Prediction precision, however, does increase with increasing number of conditioning wells, as shown by a decrease in coefficients of variation and an increase in the volume fraction of the 3D block that is predicted with high precision. The coefficient of variation of total volume of channel-belt deposits decreases from 0.18 to 0.13 (Figure 7.13A). Similarly, coefficients of variation for connectedness ratio decrease from 0.45 with no conditioning wells to 0.21 when five conditioning wells are used (Figure 7.13B). Addition of more well data also increases the precision of the prediction of the 3D block of alluvial architecture. The volume fraction of this block that is predicted by the model with a high probability (> 0.9) of occurrence of either channel-belt or overbank deposits increases with increasing number of wells (Figure 7.14). If there is no conditioning to well data, a volume fraction of only 0.08 is predicted by the model as channel-belt deposits or overbank deposits with a probability of occurrence greater than 0.9. With five wells, this fraction is 0.15.

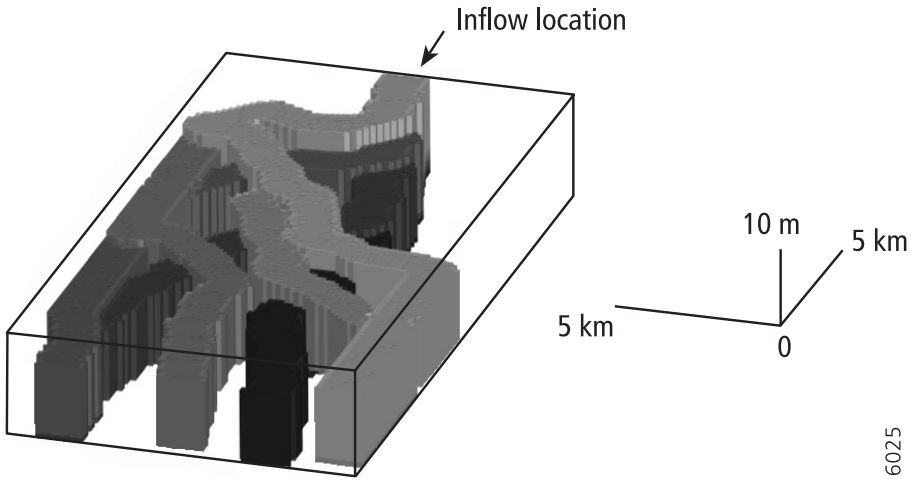


Figure 7.9. 3D alluvial architecture resulting from the realization of the model in Figure 7.7. Only channel-belt deposits are shown.

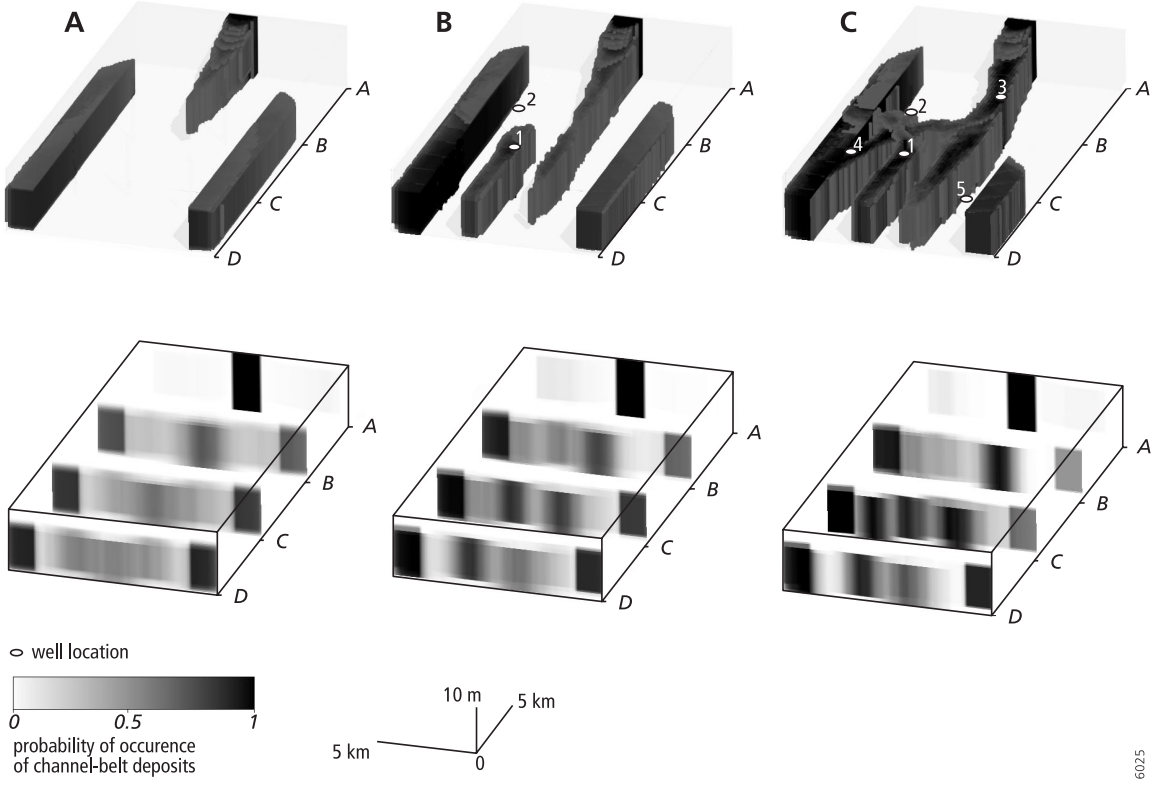


Figure 7.10. 3D images of probability of occurrence of channel-belt deposits. Top diagrams show volumes with probability > 0.8. Bottom diagrams are transects through the 3D block; grayscale represents probability of occurrence of channel-belt deposits. A) Model unconditioned to wells. B) model conditioned to wells 1 and 2. C) model conditioned to wells 1 - 5.

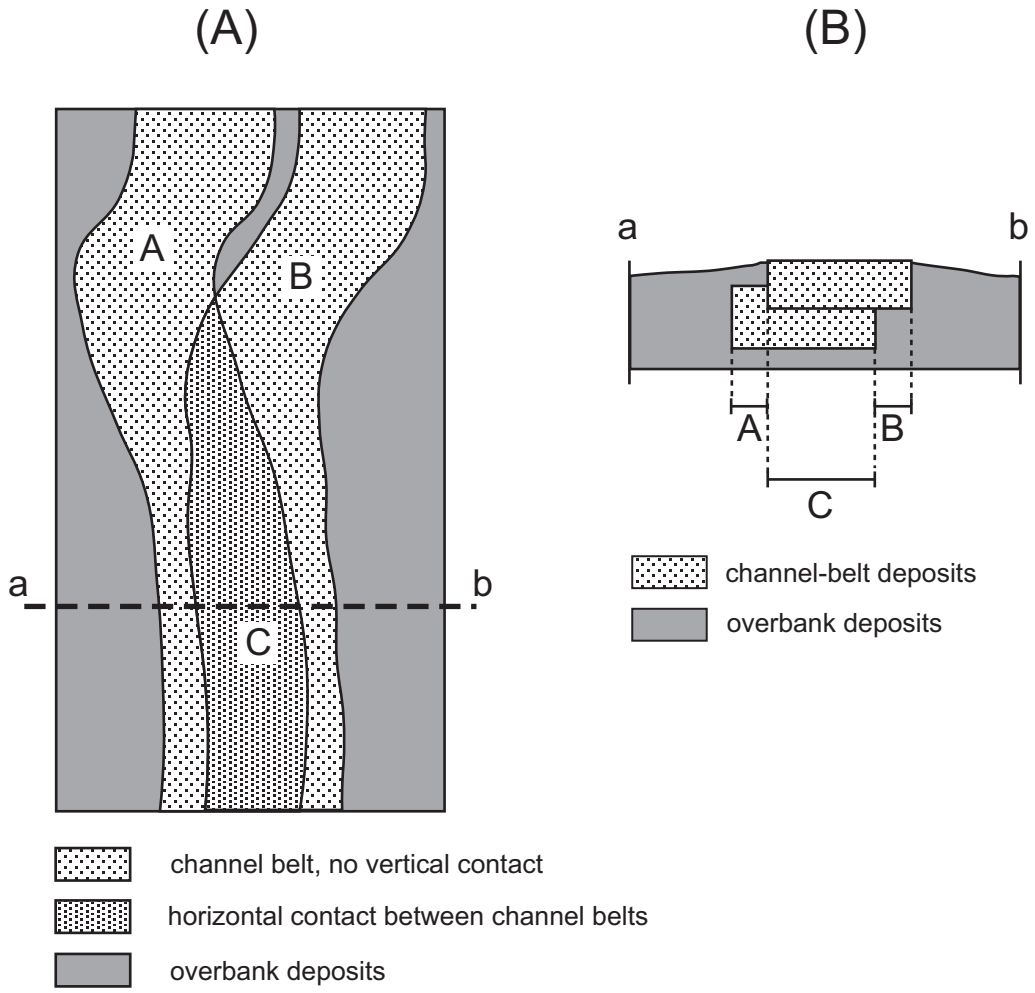


Figure 7.11. Method of calculation of areal channel-belt connectedness ratio, A) plan view, B) cross section. A, B, and C represent areas (m^2). Horizontal contact between channel belts exists in area C. Channel-belt connectedness ratio is 2C/(A+B+2C).

7.5 Discussion and conclusions

The goal of this study was to show that it is possible in principle to condition process-based models to observational data. The example given here does not prove the applicability of the process-based model to real-world problems, because the model and the hypothetical well data were relatively simple. Fitting of the model to the well data using Monte Carlo simulation required much computer time. For example, 5000 realizations took five days running time on a 200 MHz Linux machine to arrive at 50 realizations that fit five wells. As a result, success of the method with real-world problems depends on future decrease of model run times. It is expected that model run times can be reduced by development of faster computers, optimizing the computer program (e.g., improve the directive function), providing key input parameters, and

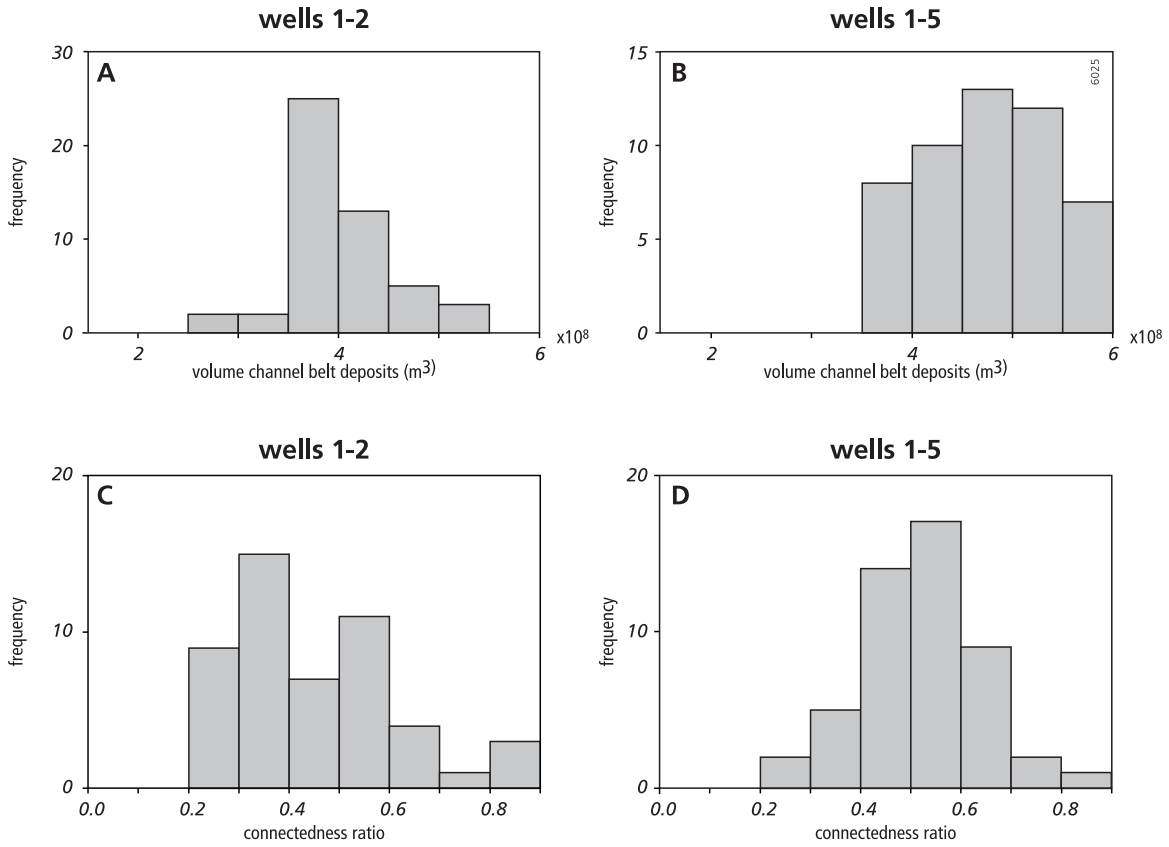


Figure 7.12. Probability density distributions for (A, B) total volume of channel-belt deposits in the well area and (C, D) channel-belt connectedness ratios in the well area. A, C) Model conditioned to wells 1 and 2; B, D) model conditioned to all the wells 1 - 5.

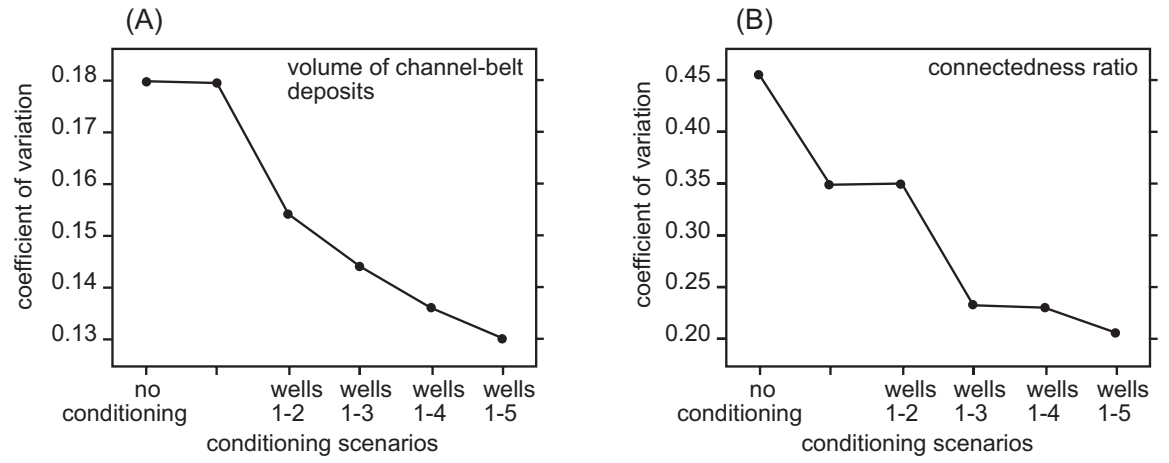


Figure 7.13. Coefficients of variation (standard deviation divided by mean value) for A) total volume of channel-belt deposits and B) connectedness ratio. Different well-conditioning scenarios on *x* axis.

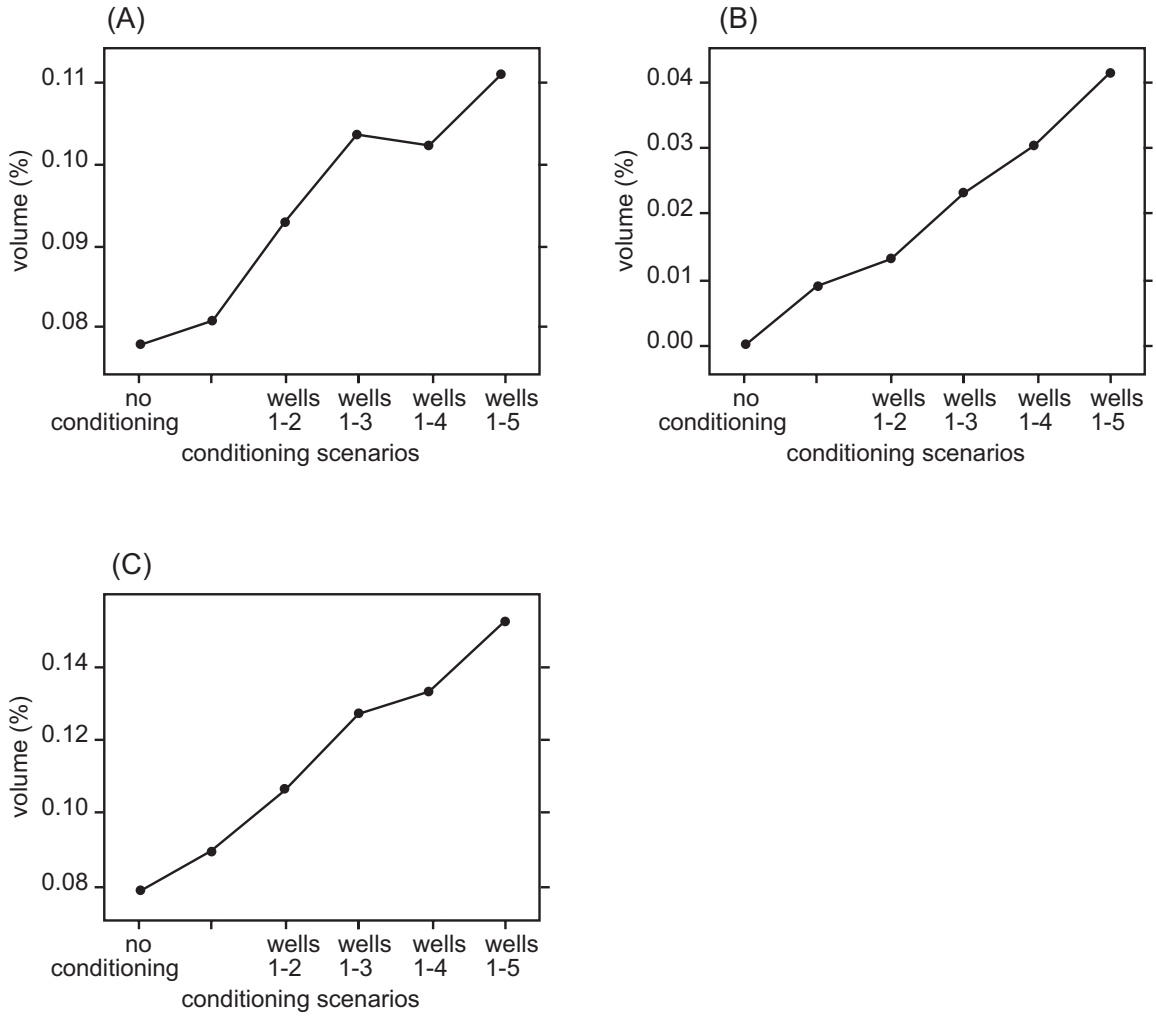


Figure 7.14. Volume fraction in the well area that is A) classified as containing no channel-belt deposits with a probability > 0.9; B) classified as containing channel-belt deposits with a probability > 0.9; and C) total volume classified with a probability > 0.9 (sum of values in graph A and B). Different scenarios of conditioning to well data on the *x* axis.

application of optimization algorithms for finding model outcomes that fit well data, such as genetic algorithms (Bornholdt et al. 1999). However, model run times will be increased as the process-based models become more complicated.

The computing time required depends on the probability that one run of the process-based model will fit the well data. Figure 7.15A shows the probability distribution U_{au} of the alluvial architecture generated by the unconditioned model and the range r of all possible alluvial architectures that fit the observational data. For illustrative purposes, U_{au} is assumed to be normal here, although other probability distributions would be possible. The probability p that one run of the process-based model fits the well data is the area under the curve of U_{au} for alluvial architectures r. The probability p depends on several factors. Increasing the number of wells for conditioning decreases the range of alluvial stratigraphies r that will fit all wells, resulting in a smaller value for p (Figure 7.15B). Increasing the width of the tolerance bands w_t in the objective function increases the

range of alluvial stratigraphies r that are regarded as being conditioned, resulting in a higher value for p (Figure 7.15C). If the process-based model or its input parameters are incorrect, r will be at one of the tails of the distribution of U_{au} and p values will be very small (Figure 7.15D) or even zero (Figure 7.15E). This can be solved by increasing the variability of the stochastic input of the model, resulting in a wider distribution of U_{as}, but p values will be relatively low (Figure 7.15F).

Successful conditioning of a process-based model depends on the amount of soft information that is available in addition to hard observational data. Soft information may come from seismic profiles and ancient or modern analogs. Conditioning to seismic data and analog data is expected to be possible with some changes in the objective and directive functions. Soft data may include paleoflow direction, channel-belt geometry, aggradation rates, floodplain width, and the presence of synsedimentary faults. If these data are not available, a process-based model can still be used by using stochastic variables for these input parameters. In the example given here, it is assumed (in the

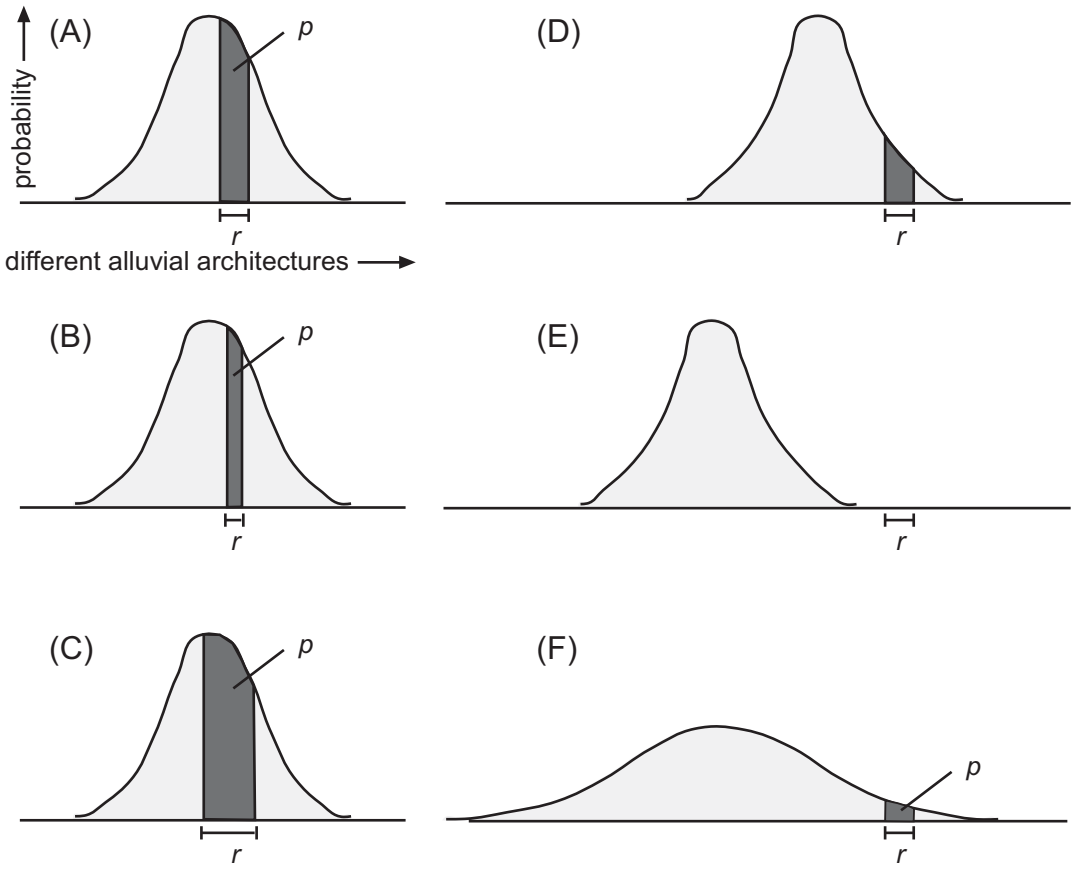


Figure 7.15. Schematic probability density distribution (U_{au}, area under the curve is 1) of alluvial architecture for unconditioned model run and range of possible alluvial architectures conditioned to well data r. Each location on the x axis represents a different alluvial architecture. The x axis is the same for all figures. The area under the curve of U_{au} for r is the probability p for a conditioned model outcome. A) Standard curve. Effects of B) larger number of well data, C) larger tolerance bands, D, E) incorrect model structure or model parameters, F) higher degree of variability in the stochastic input.

interest of simplicity) that these parameters were known except for the initial floodplain elevation, which was represented as a stochastic variable. Adding more stochastic inputs to process-based models would be a worthwhile extension to the approach of sedimentary architecture modeling described here.

Under the assumptions of a correct model structure and input, the Monte Carlo method applied here gives model outputs that are true probability distributions. For each voxel, the probability of occurrence and the connectedness of channel-belt deposits is known. This is an advantage compared to some stochastic modeling studies in which only one or a few realizations are given. Another advantage of Monte Carlo simulation is the possibility of deriving a relationship between the number of observational data (i.e., wells) and the precision of the predicted architecture.

The quality of prediction of alluvial architecture using process-based models depends strongly on the quality of the model used. Current knowledge of alluvial processes allows considerable improvement of the model presented here. Such development would be worthwhile in view of the potential of this method of fitting process-based models to observational data. Furthermore, the potential use of process-based models to provide input to purely stochastic models would also require further development of process-based models.

Process-based models conditioned to observational data could potentially be used in other depositional environments such as coastlines, marine shelves, and submarine fans. The method also has potential for application at larger scales (such as sequence-stratigraphic models) and smaller scales (such as crevasse-splay models).

Acknowledgments

We thank Cees Wesseling (PCRaster Environmental Software), Kor de Jong, and Edzer Pebesma (Utrecht University) for programming and providing the gstat and PCRaster software used for the numerical simulations in this chapter; and Jaap Kwadijk for providing the initial impetus for sedimentary architecture modeling in PCRaster. We are also grateful to Marc Bierkens, Peter Burrough, and Kees Geel, as well as journal referees Colin North, Paul Heller, and Jeffrey Yarus for helpful comments on earlier drafts.

7.6 References

Anderson, M.P. (1997), Characterization of geological heterogeneity, in Dagan, G., and Neuman, S.P., eds., Subsurface Flow and Transport: A Stochastic Approach: Cambridge, U.K.: Cambridge University Press, p. 23-43.

Bierkens, M.F.P. & Weerts, H.J.T. (1994), Application of indicator simulation to modelling the lithological properties of a complex confining layer. Geoderma 62, pp. 265-284.

Bornholdt, S., U. Nordlund & H. Westphal (1999), Inverse stratigraphic modeling using genetic algorithms, in Harbaugh, J.W., Watney, W.L., Rankey, E.C., Slingerland, R., Goldstein, R.H., and Franseen, E.K., eds., Numerical Experiments in Stratigraphy: Recent Advances in Stratigraphic and Sedimentologic Computer Simulations, SEPM, Special Publications 62, p. 85-90.

Bridge, J.S. (1977), Flow, bed topography, grain size and sedimentary structure in open channel bends: a three-dimensional model. Earth Surface Processes 2, pp. 401-416.
Bridge, J.S. (1992), A revised model for water flow, sediment transport, bed topography, and grain size sorting in natural river bends. Water Resources Research 28, pp. 999-1013.
Bridge, J.S. & M.R. Leeder (1979), A simulation model of alluvial stratigraphy: Sedimentology 26, pp. 617-644.
Bryant, I.D. & S. Flint (1993), Quantitative clastic reservoir geological modeling: problems and perspectives. In Flint, S. and Bryant, I.D., eds., The Geological modelling of hydrocarbon reservoirs and outcrop analogues: International Association of Sedimentologists, Special Publication 15, p. 3-20.
Budding, M.C., A.H.M. Paardekam & S.J. van Rossem (1992), 3D connectivity and architecture in sandstone reservoirs. SPE Technical Paper 22342, presented at SPE International Meeting on Petroleum Engineering, Beijing, China, 24-27 March.
Burrough, P.A. & R.A. McDonnell (1998), Principles of Geographical Information Systems. Oxford, U.K.: Oxford University Press.
Carle, S.F., E.M. Labolle, G.S. Weissmann, D. van Brocklin & G.E. Fogg (1998), Conditional simulation of hydrofacies architecture: a transition probability/Markov approach, in Fraser, G.S., and Davis, J.M., eds., Hydrogeologic Models of Sedimentary Aquifers: SEPM, Concepts in Hydrogeology and Environmental Geology No. 1, pp. 147-170.
Clemetsen, R., A.R. Hurst, R. Knarud & H. Omre (1990), A computer program for evaluation of fluvial reservoirs, in Buller, A.T., Berg, E., Hjelmeland, O., Kleppe, J., Torsaeter, O., and Aasen, J.O., eds., North Sea Oil and Gas Reservoirs-II: London, Graham & Trotman, pp. 373-385.
Cross, T.A. & M.A. Lessenger (1999), Construction and application of a stratigraphic inverse model, in J.W. Harbaugh, W.L. Watney, E.C. Rankey, R. Slingerland, R.H. Goldstein & E.K. Franseen, eds., Numerical Experiments in Stratigraphy: Recent Advances in Stratigraphic and Sedimentologic Computer Simulations, SEPM, Special Publications 62, p. 69-83.
Deutsch, C. & P. Cockerham (1994), Practical considerations in the application of simulated annealing to stochastic simulation. Mathematical Geology 26, pp. 67-82.
Deutsch, C.V. & L. Wang (1996), Hierarchical object-based stochastic modeling of fluvial reservoirs: Mathematical Geology 28, pp. 857-880.
Doveton, J.H. (1994), Theory and applications of vertical variability measures from Markov Chain analysis In J.M. Yarus & R.L. Chambers, eds., Stochastic Modeling and Geostatistics: American Association of Petroleum Geologists, Computer Applications in Geology, no. 3, pp. 55-64.
Hammersley, J.M. & D.C. Handscomb (1979), Monte Carlo Methods. London: Chapman & Hall.
Heller, P.L. & C. Paola (1996), Downstream changes in alluvial architecture: an exploration of controls on channel-stacking patterns. Journal of Sedimentary Research 66, pp. 297-306.
Heuvelink, G.B.M. (1998), Error Propagation in Environmental Modelling with GIS. London: Taylor & Francis.
Hirst, J.P.P., C.R. Blackstock and S. Tyson (1993), Stochastic modelling of fluvial sandstone bodies, in S.S. Flint & I.D. Bryant, eds., The Geological Modelling of Hydrocarbon Reservoirs and Outcrop Analogues: International Association of Sedimentologists, Special Publication 15, pp. 237-252.
Holden, L., R. Hauge, Ø. Skare & A. Skorstad (1998), Modeling of fluvial reservoirs with object models. Mathematical Geology 30, pp. 473-496.
Horn, B.K.P (1981), Hill shading and the reflectance map: Institute of Electrical and Electronics Engineering, Proceedings 69, pp. 14-47.
Journel, A.G. (1983), Nonparametric estimation of spatial distributions: Mathematical Geology 15, pp. 445-468.

Gross, L.J. & M.J. Small (1998), River and floodplain process simulation for subsurface characterization. Water Resources Research 34, pp. 2365-2376.

Koltermann, C.E. & S.M. Gorelick (1996), Heterogeneity in sedimentary deposits: A review of structure-imitating, process-imitating, and descriptive approaches. Water Resources Research 32, pp. 2617-2658.

Mackey, S.D. & J.S. Bridge (1995), Three-dimensional model of alluvial stratigraphy: theory and application. Journal of Sedimentary Research B65, pp. 7-31.

Moore, I.D. (1996), Hydrological Modeling and GIS. In M.F. Goodchild, L.T. Steyaert, B.O. Parks, C. Johnston, D. Maidment, M. Crane & S. Glendinning, eds., GIS and Environmental Modeling: Progress and Research Issues: Fort Collins, Colorado, GIS World Books, pp. 143-148.

North, C.P (1996), The prediction and modelling of subsurface fluvial stratigraphy. In P.A. Carling and M.R. Dawson, eds., Advances in Fluvial Dynamics and Stratigraphy. New York: John Wiley & Sons Ltd, p. 395-508.

Pebesma, E. & C.G. Wesseling (1998), Gstat: a program for geostatistical modelling, prediction and simulation. Computers & Geosciences 24, pp. 17-31.

Slingerland, R. & N. Smith (1998), Necessary conditions for a meandering-river avulsion. Geology 26, pp. 435-438.

Stam, J.M.T. (1996), Migration and growth of aeolian bedforms. Mathematical Geology 28, pp. 519-536.

Tetzlaff, D.M. & J.W. Harbaugh (1989), Simulating Clastic Sedimentation. New York: Van Nostrand Reinhold.

Tyler, K., A. Henriquez & T. Svanes (1994), Modeling heterogeneities in fluvial domains: a review of the influence on production profiles. In J.M. Yarus & R.L. Chambers, eds., Stochastic Modeling and Geostatistics: American Association of Petroleum Geologists, Computer Applications in Geology 3, pp. 77-89.

Van Deursen, W.P.A. (1995), Geographical Information Systems and Dynamic Models. Utrecht: Koninklijk Nederlands Aardrijkskundig Genootschap/Faculteit Ruimtelijke Wetenschappen, Universiteit Utrecht.

Webb, E.K. (1994), Simulating the three-dimensional distribution of sediment units in braided-stream deposits. Journal of Sedimentary Research B64, pp. 219-231.

Wesseling, C.G., D. Karssenberg, W.P.A. van Deursen & P.A. Burrough (1996), Integrating dynamic environmental models in GIS: the development of a Dynamic Modelling language. Transactions in GIS 1, pp. 40-48.

PART IV:

TRAINING

8 THE PCRASTER SOFTWARE AND COURSE MATERIALS FOR TEACHING NUMERICAL MODELLING IN THE ENVIRONMENTAL SCIENCES

Derek Karssenberg (with Peter A. Burrough, Raymond Sluiter, Kor de Jong)

Abstract: Teaching numerical modelling in the environmental sciences not only needs good software and course material but also an understanding of how to program the models in the computer. Conventional environmental modelling procedures require computer science and programming skills, which may detract from the important understanding of the environmental processes involved. An alternative strategy is to build a generic toolkit or modelling language that operates with concepts and operations that are familiar to the environmental scientist. PCRaster is such a spatio-temporal environmental modelling language developed at Utrecht University, the Netherlands. It is used for teaching modelling in classrooms and over the WEB (distance learning) at three levels: 1) explaining environmental processes and models, where models with a fixed structure of model equations are evaluated by changing model parameters, 2) teaching model construction, where students learn to program spatial and temporal models with the language, and 3) teaching all phases of scientific modelling, related to field research. So far, we have received positive responses to these courses, largely because the software provides a set of easily learned functions matching the conceptual thought processes of a geoscientist that be used at all levels of teaching.

Reproduced from: Karssenberg, D., P.A. Burrough, R. Sluiter & K. de Jong (2001). The PCRaster software and course materials for teaching numerical modelling in the environmental sciences. Transactions in GIS 5, pp. 99-110.

8.1 Introduction

As in many areas of science, numerical modelling is a means of expressing our understanding of complex, dynamic processes, such as those operating in natural or semi-natural environments. Not so long ago, it was sufficient for environmental scientists merely to name the forms created by frequently occurring interactions of many processes - soil series, vegetation types, climatic zones: then classification was an aim in itself. Today, modelling has largely replaced classification as a scientific activity as scientists attempt to find better and more quantitative explanations of how complex environmental systems work. In this respect, creating a numerical model is one way of expressing our scientific understanding in a communicable form. It is therefore complete natural, and important that we should want to teach numerical modelling to our students.

Environmental systems are mostly open systems with many, poorly definable interactions between hard to describe processes. These complicated systems having both spatial and temporal variation at several levels of resolution are difficult to reproduce in the laboratory. Since analogue modelling requires large scale models, it is generally not

feasible, especially for teaching, so numerical methods are currently most used for modelling environmental systems. Whereas engineering skills and tools are needed to build an analogue model, mathematical and computer programming skills and tools are necessary to build numerical models. Today, many environmental scientists have access to powerful computers, but they are not necessarily able to program them. Until recently, the creation of complex environmental models has required the efforts of teams of computer scientists - think of MODFLOW (McDonald and Harbaugh 1984) and related pollutant transport codes. This is mainly because such models need complex numerical algorithms, programmed by specialist programmers in system programming languages such as C or Fortran. For teaching environmental model construction, the use of system programming languages for constructing a model presents difficulties because many students in the geosciences are not skilled in programming computers. Instead, a modelling language is needed that matches the thinking level of an environmental researcher or geoscientist who thinks in terms of higher level functions that may be used to simulate real world processes.

A second issue in education is the computing load. It is not uncommon for some environmental models to require super-computers for their operation. While this is appropriate for highly specific problems, such methods are out of the question for routine modelling and instruction in class (Casti 1998). Clearly, there is a need for a generic environmental modelling language that operates in a way that is easily understandable for students and environmental scientists providing flexible, modular model construction on affordable computers. PCRaster is such an environmental modelling language (Van Deursen 1995; Wesseling et al 1996). It is raster-based and supports construction of both static GIS-like cartographic models and dynamic spatio-temporal models (including cellular automata functions) using a modelling syntax similar to mathematical notation.

This chapter explains how the PCRaster software may be used for teaching numerical modelling in a classroom situation and for distance learning (Web based teaching). At Utrecht University, course material has been written for both kinds of teaching environments, which supports individual learning without too much direct contact with the tutor. This absence of direct assistance by a tutor is crucial in Web based teaching since learning over the Web does not allow direct assistance from a tutor. Instead, support has to be provided by the course material itself. The chapter describes methods how this automatic assistance is provided as an integral part of the course material.

The Utrecht modelling curriculum has a learning path that starts with map overlay operations and static cartographic modelling (c.f. Burrough and McDonnell 1998), and simple numerical modelling for the construction and application of spatio-temporal environmental models (Burrough 1998, Wesseling et al 1996). Geoscientific fields that use such spatio-temporal modelling include hydrology, ecology, pedology, sedimentology, land degradation research and geomorphology. The students are undergraduate, or postgraduate students in these sciences; they are usually unfamiliar with programming, but most have some basic knowledge of GIS. At MSc level and above, students are offered courses in using PCRaster to construct models of environmental processes that they encountered in their major fieldwork course. One of the main principles behind all the material is that it should stimulate students to construct conceptual frameworks that can support the development of their own models, thereby finding solutions for their own specific problems.

8.2 The PCRaster Environmental Modelling Software

The effort and degree of specialism needed to create powerful environmental models has lead to model building being seen as a scientific activity in its own right. The products of this work - the models - may be completely understood by their creators, but others often see them as 'black box' units in a data processing system. Such models are frequently linked to GIS as sources of data, and as a means of displaying the results or for combining them with other spatial information.

This loose coupling of GIS and off-the-shelf models may be ideal for many environmental scientists working in advisory or dedicated research institutes but it poses several problems for those wishing to experiment with modelling, or to teach it to university and post-academic students. These problems include:

- The afore-mentioned need for programming skills in a system progamming language such as FORTRAN or C++
- A lack of insight into how mathematical representations of processes are expressed in algorithms in the standard models
- The difficulties or impossibilities of easily modifying the model code to explore new ways of modelling the process being studied
- A complete absence of a generic approach (apart from the mathematical basis) to dynamic spatial modelling because each program is written by a different team under different circumstances
- The need for means of linking models with spatial and temporal data in a GIS

In recent years there have been developments in both computational mathematics and GIS that provide answers to these problems. The first edition of the 'Numerical Recipes' books (Press et al. 1986) was a major advance that made many standard algorithms available to those without knowledge of numerical mathematics. But modellers still needed to be programmers, since the algoritms were still written in system programming languages. The main disadvantage of model construction in a system programming language, however, is that all technicalities of a model have to be defined in the code, resulting in difficult to handle model programs. So there is a need for a higher level programming language with standard easy to use functions at the level of thinking of a modeller, that enables the representation of real world processes. This line of reasoning has been followed by programs such as Matlab (2000) and STELLA (2000) in which the user can set up and solve equations without any overt programming skills. Such programs are in effect higher level programming languages that operate with an interface at the level of scientific or technical communication.

In GIS there is a long tradition of using of command languages for stringing together sets of well-defined operations to achieve given results. The command syntax of such languages is frequently of the form

$$Newmap = \textbf{operation}(Inputmap_{1..n})$$

in which the resulting map is created by a certain operation on one or more input maps. The **operation** is the name of a generic process that the user understands at his or her level of working. The operation could be that of calculating slope from an elevation map, or of determining proximity zones to linear features such as roads or rivers (c.f. Burrough and McDonnell 1998), so the 'language' is very easy to use. By linking these operations together one can create what are in effect 'models', albeit static models of spatial patterns that are the bread-and-butter of much GIS analysis.

This concept of higher level operations has been extended in PCRaster to include not just spatial operations but also temporal iterations. In addition the set of spatial functions in the language is increased to cover many standard operations encountered in environmental studies. The result is a mathematical space-time modelling language that gives many advantages for environmental modelling and those wishing to teach it. Data entities in PCRaster are objects such as stacks of raster maps for spatio-temporal attributes, time series for temporal non-spatial data and lookup tables. The language contains 125 generic functions operating on these entities. Functions included are mathematically defined non-spatial (point) operations, network creation, transport and flow operations and window operations, as well as spatio-temporal time operations for reading and writing temporal data, e.g. hydrographs at specific locations. A GIS-like database and visualization software is integrated in the system.

8.3 Levels of Teaching

PCRaster is used at several levels in teaching environmental modelling:

- As a means of explaining environmental processes and models
- As a modelling language in its own right and as means of teaching model construction
- As a means of teaching all phases of scientific modelling needed for field research

8.3.1 Explaining Environmental Processes and Models

At the introductionary level, students do not yet have the skills needed for constructing models. Teaching at this level has two aims. First, it is important to give students insight in processes occurring in the natural world. In a lecturing environment, models can be used as virtual landscapes illustrating processes occurring in a real landscape and they supplement – but not replace! - field excursions and field studies. Model animations have been shown to be very useful for illustrating processes that are too slow or imperceptible to experience in the field, e.g. tectonic uplift or groundwater flow. In addition, models are useful for illustrating different scenarios of landscape evolution, for example, exploring the degrees of sedimentation and erosion that may occur if a dam is built.

Secondly, at a somewhat higher level, students can run models themselves by modifying certain model parameters through a graphical user interface (GUI). By clicking on the buttons in the user interface, the user invokes the simulation software under user

defined scenarios or the standard visualization routines of PCRaster. This can be done without direct support of a lecturer, if good course material is provided. This helps students understand how landscape processes are represented by environmental models and how model components interact.

One of the Utrecht courses at this introductory level covers the long-term effects of slope processes on slope profile form. The temporal change in elevation for each location on a hillslope is a result of interaction between processes of weathering and supply or removal of material over the slope, driven by several factors, see Figure 8.1 (cf. Ahnert 1987). A PCRaster model simulating the development of a mountain range in geological times is used to explain these often difficult to understand interactions (Figure 8.2). The student learns the basic principles of hill slope processes by changing the input parameters in the interface, such as base level lowering, levels of creep and wash, and comparing animated maps and time series for different scenarios. A course text is written that stimulates students to learn by 'playing' with the model.

An advantage of using PCRaster at this level of teaching is that lecturers can quickly and easily make models with student interfaces that will run on personal computers. PCRaster automatically generates the GUI from a PCRaster modelling script written by the lecturer, without extra programming. As a result, development times of models are short, and models can be tailored to a specific lecture or theme. In addition, evaluating models by interactively changing model parameters requires the students to think, and inspires them to make their own models.

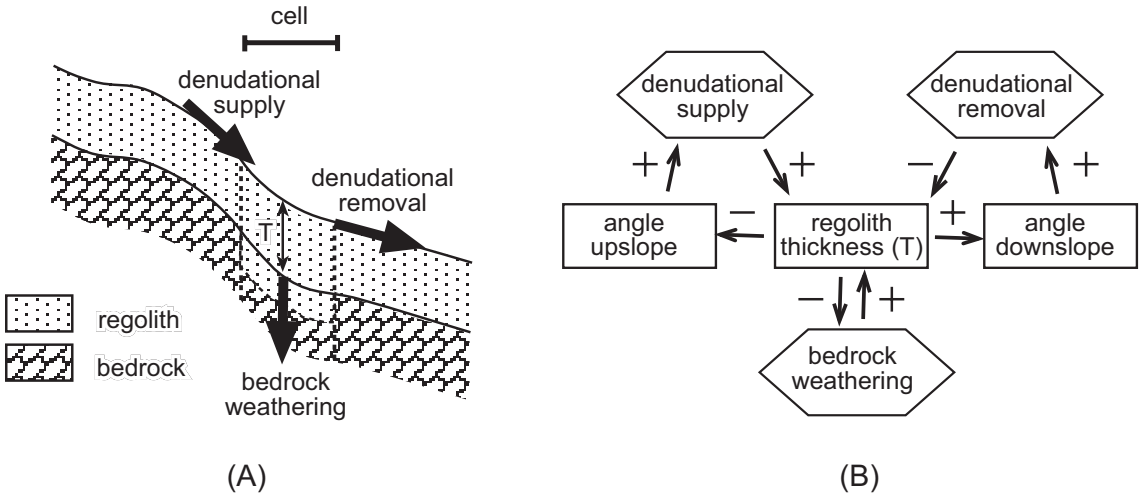


Figure 8.1. Factors and processes determining the rate of bedrock weathering (i.e. denudation) at a grid cell on a hillslope, (A) profile with processes, (B) positive and negative feedback loops in the cell. After Ahnert, 1987.

8.3.2 Teaching Model Construction

At this second level, students are taught how to construct simple models using PCRaster. They learn how the language is used and construct models step by step. The course text gives the basic theory behind the models and explains how the models may be constructed with PCRaster. This level is taught in two phases: static modelling and dynamic modelling.

8.3.2.1 *Static Modelling.* Static modelling involves modelling in space with single PCRaster operations or combinations of operations in script files. The student learns the ideas behind the different groups of (non-spatial) point operations and spatial operations by applying them to a case study data set. Initially, students work from the command line, by typing single operations with a syntax similar to mathematical notation. For instance, runoff during a rainstorm is calculated by:

```
pcrcalc RunoffMap = accuthresholdflux(LddMap,RainMap,InfilMap)
```

where `LddMap` contains the flow directions, and `RainMap`, `InfilMap` contain the amount of rain and potential infiltration respectively, under the assumption of steady state conditions (Burrough 1998, Wesseling et al. 1996, van Deursen 1995 - Figure 8.3, p. 196). In the next phase, students have to solve case study problems by combining operations in a PCRaster script file. Here it is not just the meaning of the single operation that counts, but also how functions can be combined, where the output of an operation can be the input of the next operation. Examples of problems which may be solved are the location of a sports fields complex in an urban area or finding the optimal route for a road through a mountainous terrain.

8.3.2.2 *Dynamic Modelling.* Dynamic modelling builds upon static modelling with script files that include the time component. The construction of dynamic models is done in a PCRaster script file structured in different sections containing PCRaster operations written using the same syntax as in static modelling (Figure 8.4). Operations in the initial section define the initial state of the model at time step = 0. In an event-based runoff model, the initial section could define the initial soil moisture content, derived by crossing a soil type map with a lookup table of soil types and their hydrologic characteristics. The temporal behaviour is defined in the dynamic section. The same set of operations in this section is carried out for each time step. The results from the previous time step are the input for the next time step. For example, in a runoff model, the dynamic section would contain an iterative operation that for each time step and each cell, adds the amount of infiltrated water to the soil moisture content. This soil moisture map could be used in the next time step for calculating potential infiltration. Additionally, the dynamic section contains time operations that read temporal data such as rain time series or cloud cover maps from the database. Storing results is simply done by adding a 'report' keyword to operations that should be stored in the database. The results can be visualized as 4D (3D plus time) displays (Figure 8.5, p. 197). The current course material includes exercises for constructing large scale catchment models, event based runoff modelling, plant growth and dispersion modelling and pollutant dispersion modelling.

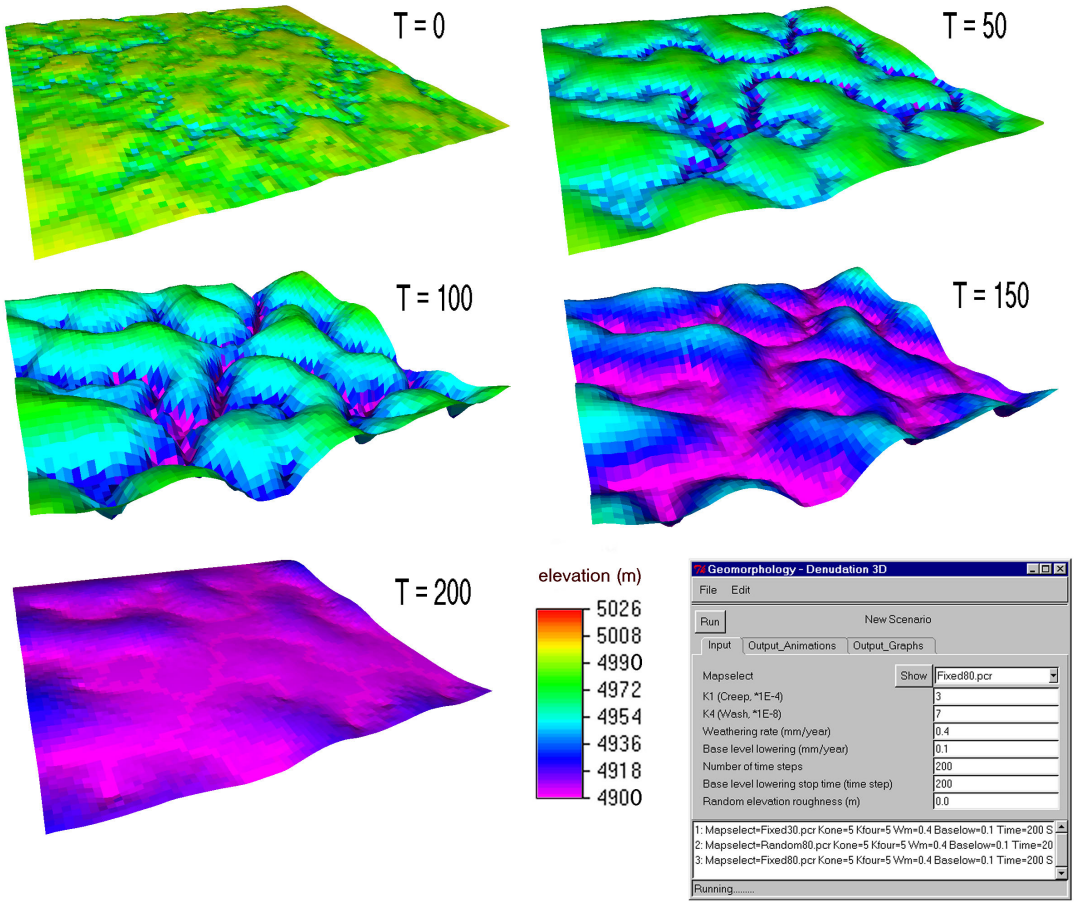


Figure 8.2. Slope development model, interface and model output at different time steps T. The output represents a baselevel lowering between T=0 and T=100, with erosion resulting in a decrease in elevation mainly near the streams and incision of valleys. After T=100, baselevel is constant and existing relief disappears as a result of continuous erosion at the higher parts and a constant level near the streams.

Our experience in Utrecht with these PCRaster-based courses suggest that PCRaster is an ideal environment for teaching model construction because:

- Its generic approach to modelling using a wide range of standard algorithms and functions avoids having to re-invent the wheel
- There is no need for students to learn a programming language such as FORTRAN or C++ before they start to think quantitatively about landscape change: if they wish, they can do this in a later stage of their study
- The modelling language is similar to that of scientific discourse – therefore it is easy to learn
- Its text-based interface supports both simple static modelling with single commands and the construction of space-time models with the same set of operations

- The language provides a means to embed modelling in the GIS rather than create once-off or clumsy links between models and GIS
- The language is portable and can be run on different platforms, including currently available, cheap PCs

As a result, the learning path is fairly short: after two weeks students are able to use the language without too many problems and are ready for the next stage of environmental modelling: the construction of models according to model concepts tailored to their own field research.

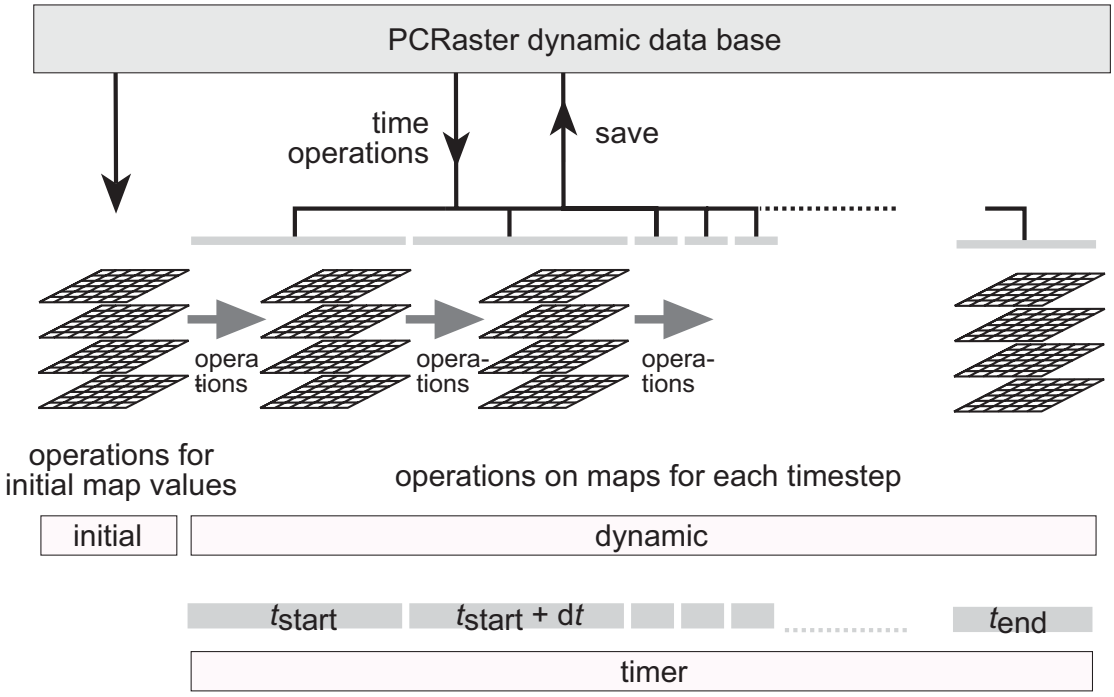


Figure 8.4. Concept of a dynamic modelling script with initial, dynamic and timer sections.

8.3.3 Teaching all Phases of Scientific Modelling Related to Field Research

At this level, equivalent to MSc and above, students perform a complete study by combining data gathered at their own field study site and knowledge of environmental processes with modelling. Their modelling study includes all phases of scientific modelling:

- identification of the structure of a model based upon the field study problem and available data, i.e. choice of processes to be simulated, mathematical representation of these processes, and the levels of temporal and spatial resolution,
- implementation of this model with PCRaster,
- calibration and validation with their own field data, and

- presentation of the results.

Students get some assistance from their tutor, especially in the first phase of model structure identification, but most of the work is accomplished without much support. Some studies performed by undergraduate students include event-based runoff and erosion modelling studies, water balance, vegetation growth and grazing modelling studies, heat balance modelling studies using remote sensing, and modelling of geomorphological processes.

Unlike the use of standard models in student research, requiring students to construct their own models has the advantage that they have to identify model structures themselves, which forces them to understand a landscape in a numerical form. In addition, students are directly confronted with the usual problems of modelling such as incorrect model structures, implementation errors, the effects of spatial and temporal resolution, and parameter identification problems, since they are responsible for everything. This responsibility is also the reason for their appreciation of this work: it is much more interesting and challenging to construct your own model, using your own field data, than pressing buttons in a model written by somebody else. A disadvantage of students constructing their own models is that they may use inappropriate concepts and methods, but this can be prevented by good support from the tutor. Another disadvantage is that the restricted set of standard functions in PCRaster limits the range of models, but this can usually be solved by programming extra functions in C++, which are called up by the plug-in function of PCRaster. This may help to give students some knowledge of system programming as well.

8.4 Courses in Classrooms and Distance Learning

8.4.1 Structuring the Course Material

Our course material for teaching in a classroom and over the internet includes a software manual and a course text for teaching at different levels of environmental modelling. We have developed the material in a structured way resulting in a database with all written material based on the Extensible Markup Language (XML, 2000). This approach has several advantages over developing course material as separate entities:

- Modularity and reuse of pieces of text. As the same operations may be repeated in different courses or different parts of a course, standard texts have been written explaining these operations which can be included at any location in the course material. This prevents double work.
- Links between courses. By using one structured database of all written material, it is easy to include and maintain hyper links within or between different courses and the user manual. This increases the accessibility of the specific information a student needs while making the exercises.

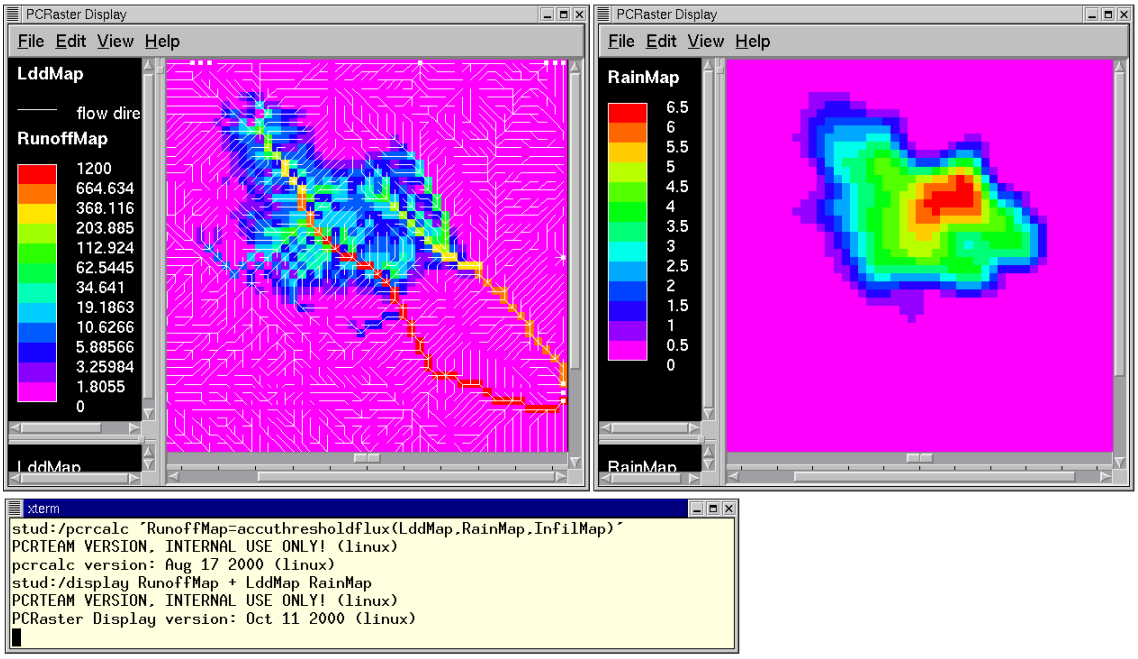


Figure 8.3. Runoff calculation with the **accuthresholdflux** operator. Right: rainstorm; left, resulting runoff.

- Recombining courses. Post-graduate students need courses tailored to their individual needs. A structured database allows course material to be extracted to meet specific learning paths.
- Standard output format for digital and hardcopy material. A standard style sheet is used after the text has been written resulting in a standard format for all material. In addition, it is possible to derive both digital (e.g. HTML) and hardcopy output from the same course text, with a standard format.

8.4.2 Distance learning

People working at government institutes and private firms may have little chance to spend periods away from home to study. These people cannot attend courses at a university campus but need courses that can be followed part time. Such on-job training courses can be provided by web based training, or distance learning.

Distance learning courses are different from courses on the campus in the sense that apart from an initial seminar, there is little direct contact between the tutor and the student. Instead, all contact is through the internet (formerly the mail), mostly with time lags. PCRaster has been incorporated in a distance learning course system that is currently developed in the EU-sponsored MUTATE bundle (MUTATE 2000). The course database contains course content in XML, written by multiple authors, and a course management tool managing and storing all the data per student, such as subscription details per module, and scores on exercises (Figure 8.6). The students retrieve and navigate through the course material on a WWW server, over the internet. Since spatio-temporal data sets may be too large to be sent over the internet, PCRaster runs with datasets on the student's computer.

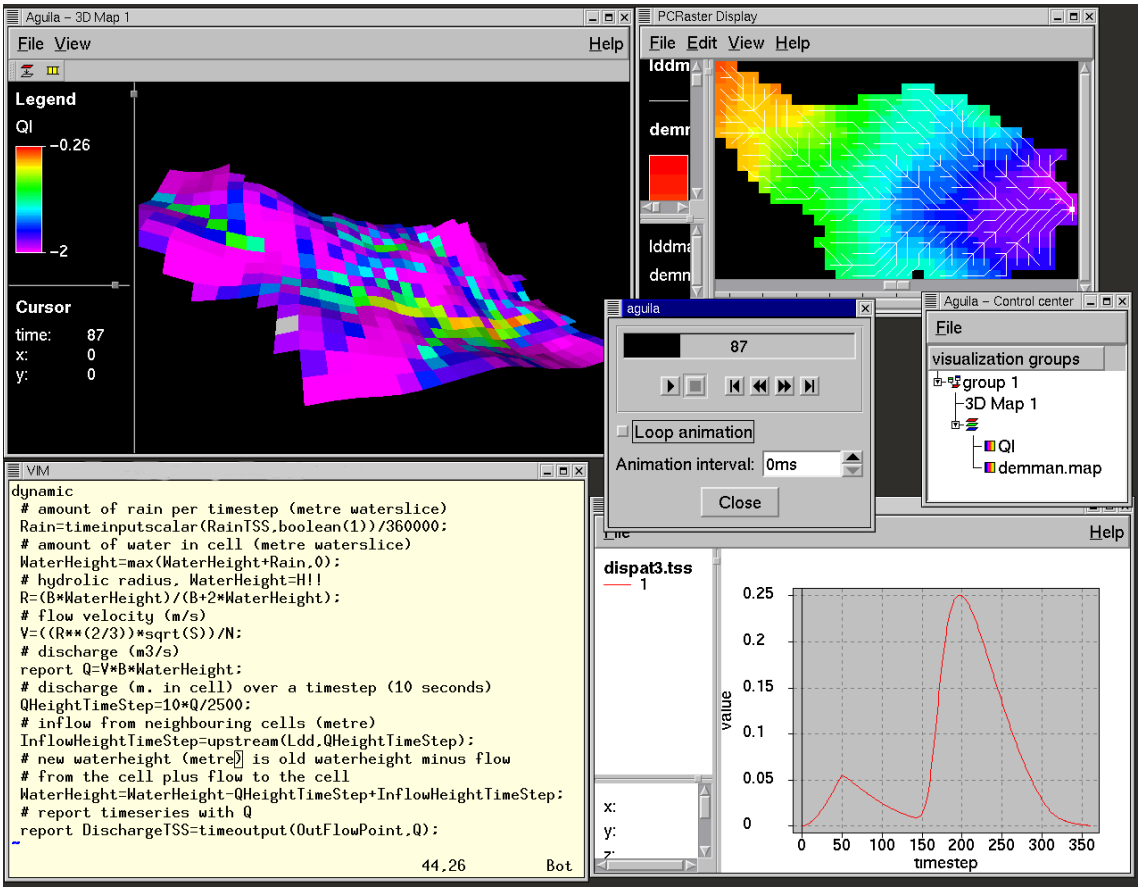


Figure 8.5. Dynamic modelling environment for the construction of an event based runoff model. Clockwise from top-left: 3D display with animation of discharge draped over the elevation model, elevation model with drainage pattern, control centre of visualisation tools, hydrograph at catchment outlet, modelling script. Centre: control centre for animations.

Since there is little direct contact between the student and tutor, the system provides alternative ways for student evaluation and support. The course material includes multiple choice, numerical and keyword questions. Students enter their answers in the digital course material and get an automatic feedback on their answer providing additional explanation, and a score, which is also used for final evaluation of the student by the tutor. This automatic evaluation is possible since the course material contains algorithms to compare students' answers with correct answers available in the course data base. In addition, open questions are given for more complicated problems, which are evaluated by the tutor. The system also stores a log of all actions of the student with the PCRaster software in the database. This information can be used by the tutor to check what the student does and provides a good reference for communication between the tutor and the student if the student has problems solving the exercises.

For courses at the second and third level that teach the construction of models, support from the tutor is very important, since students may easily get stranded while constructing a model. In a classroom situation, this support is provided by a tutor in the

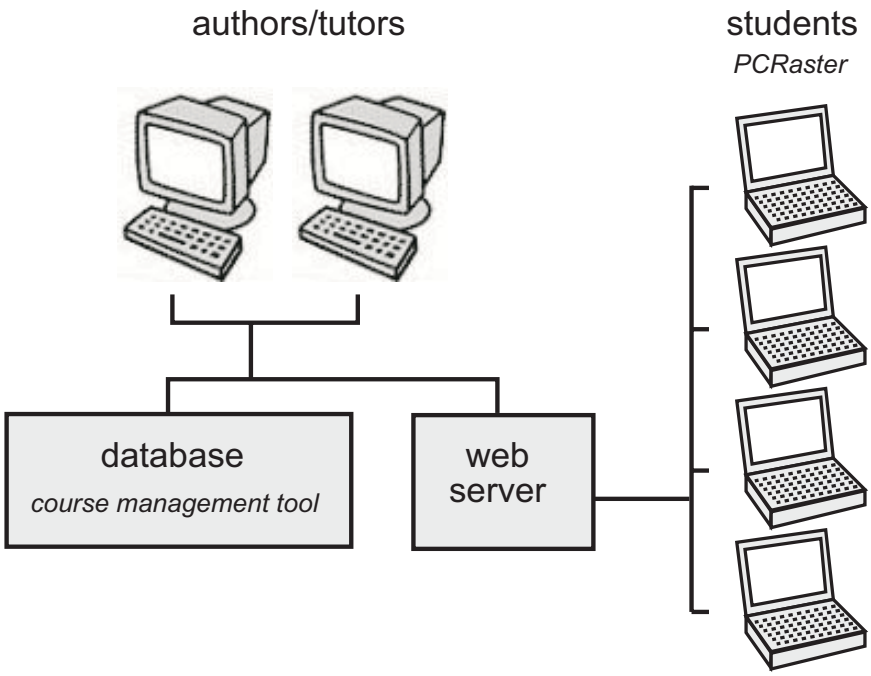


Figure 8.6. Distance learning environment.

classroom. Distance learning needs alternative strategies to provide comparable support. Strategies followed in the PCRaster course material are:

- Provide the student with background knowledge at the start of each module. This is done by referring to pages in standard text books, links to the PCRaster manual and internet sites.
- Provide links to (parts of) courses, which the student has already done, to freshen up the skills needed for a specific exercise.
- Provide advice on the basis of answers to questions, which are automatically evaluated. If the student gives the wrong answer, the system explains why this answer is wrong and provides (parts of) the correct solution.
- Provide optional links to hints giving additional information that should help the student solve the problem.
- Provide methods for contacting other students, such as bulletin boards or mailing lists.
- Provide means to contact the tutor by email attaching all additional information (model scripts, commands) that may help the tutor to support the student.

8.5 Discussion and Future Work

Although the existing software and course content provides modelling courses in a wide range of fields, PCRaster does not support all types of modelling which could be the

subject of teaching in one of the geosciences. Geostatistical modelling, interpolation and simulation is currently supported by the Gstat package (Pebesma and Wesseling 1998), but this package is only loosely linked to PCRaster through the same raster file structure. A useful combination of modelling and geostatistics would provide methods for error propagation in spatial-temporal models, assuming spatially correlated errors in the inputs. Theory for error propagation modelling in GIS has been described in standard text books (e.g. Burrough and McDonnell 1998, Heuvelink 1998) but tools, which are easy to use for environmental modellers, are not generally available. The development of such a tool in PCRaster would need embedded linking of the Gstat random field simulation software with the PCRaster environmental modelling software (cf. Karssenberg et al. 2000; Pebesma et al. 2000).

The disadvantage that PCRaster is only a 2.5D system is currently being worked on by extending the modelling language with 3D entities and functions (Karssenberg et al. 2000). Whether the requirement for cheap computing will be met has yet to be seen, but given the still increasing processing power we expect that 3D methods will be soon available for both the classroom and research.

Acknowledgements
The authors wish to thank our colleagues in the PCRaster research team: Willem van Deursen, Cees Wesseling (PCRaster Environmental Software), Marlous van der Meer, Arko Lucieer (Utrecht University), colleagues of the Faculty of Geographical Sciences, Utrecht University and many other people who supported the development of PCRaster and associated course material through the years.

8.6 References

Ahnert, F. (1987), Process-response models of denudation at different spatial scales. Catena Supplement 10, pp. 31-50.

Burrough, P.A. (1998), Dynamic Modelling and GIS, Chapter 9, In: P. Longley et al.(Eds) Geocomputation: a Primer. New York: Wiley. pp. 165-192.

Burrough, P.A. & R.A. McDonnell (1998), Principles of Geographical Information Systems. Oxford: Oxford University Press.

Casti, J.L. (1998), Would-Be Worlds: How Simulation Is Changing the Frontiers of Science. New York: Wiley.

Heuvelink, G.B.M. (1998), Error Propagation in Environmental Modelling with GIS. London: Taylor and Francis.

Karssenberg, D., K. de Jong & P.A. Burrough (2000), A prototype computer language for environmental modelling in the temporal, 3D spatial and stochastic dimension. In WWW document, http://www.colorado.edu/research/cires/banff, paper presented at the 4th International Conference on Integrating GIS and Environmental Modelling, September 2-8, Banff, Canada.

Matlab 2000 MATLAB®6 WWW document, http://www.mathworks.com/products/matlab/

McDonald, M.G. & A.W. Harbaugh (1984), A modular three-dimensional finite-difference ground-water flow model. Reston: US Geological Survey.

MUTATE (2000), Multimedia Tools for Advance GIS Training in Europe. WWW document, http://mutate.chiron.pt/

Pebesma, E. & C.G. Wesseling (1998), Gstat: a program for geostatistical modelling, prediction and simulation. Computers & Geosciences 24, pp. 17-31.

Pebesma E.J., D. Karssenberg D & K. de Jong (2001), The stochastic dimension in a dynamic GIS. In Proceedings of Compstat 2000, 14th Conference of the International Association for Statistical Computing, 21-25 August, Utrecht, the Netherlands.

Press, W.H., S.A. Teukolsky, W.T. Vetterling & B.P. Flannery (1986), Numerical Recipes in Fortran 77, The Art of Scientific Computing (First Edition). Cambridge: Cambridge University Press.

STELLA (2000), HPS: The Systems Thinking Company. WWW document, http://www.hps-inc.com/edu/stella/stella.html

Van Deursen, W.P.A. (1995), Geographical Information Systems and Dynamic Models. Utrecht: Koninklijk Nederlands Aardrijkskundig Genootschap/Faculteit Ruimtelijke Wetenschappen, Universiteit Utrecht.

Wesseling, C.G., D. Karssenberg, W.P.A. van Deursen & P.A. Burrough (1996), Integrating dynamic environmental models in GIS: the development of a Dynamic Modelling language. Transactions in GIS 1, pp. 40-48.

XML (2000), The XML Industry Portal. WWW document, http://www.xml.org

9 Conclusions

In the introduction (Chapter 1), the important role of the model development cycle (Figure 9.1) was explained, and the central research question was given related to the model development cycle:

Is the current state of computer technology and science sufficient for executing the steps of programming, upscaling, and inverse modelling in the model development cycle, and for teaching all steps in the model development cycle, with respect to dynamic spatial environmental models simulating continuous fields?

In this section, the central research question will be answered by first treating the individual research questions related to programming, scaling, inverse modelling and teaching, which were derived from the central research question in the introduction. An answer to the central research question will be derived from these individual research questions at the end of this concluding section.

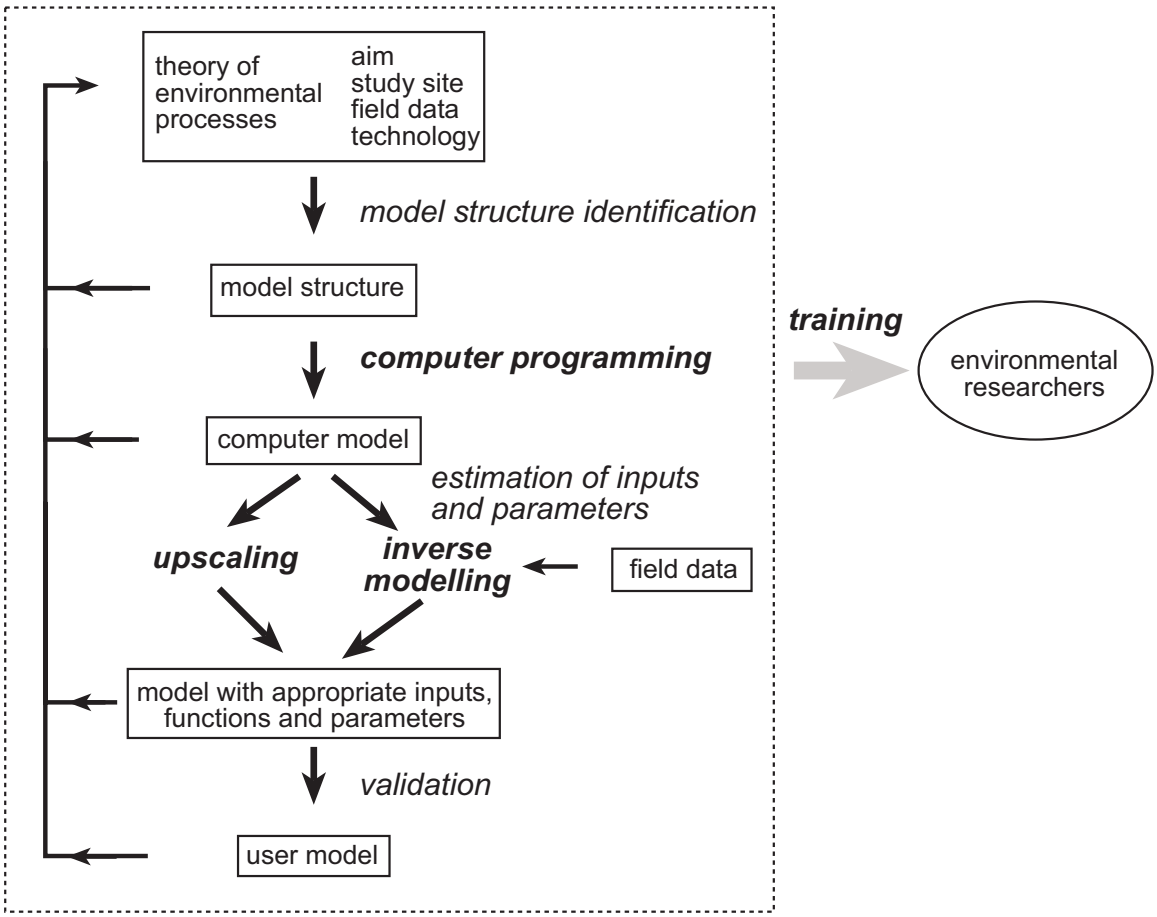


Figure 9.1. Model development cycle. Issues of programming, scaling, inverse modelling and teaching (in bold type) represent those dealt with in this thesis.

9.1 Programming

As noted in the introduction, the research questions for programming deal with the procedural step in the model development cycle comprising the conversion of the mathematical representation of the processes to a computer program of the model (Section 1.3.3). Two groups of research questions were posed which will be answered here: those related to (concepts of) existing dynamic spatial environmental modelling languages, and those dealing with scientific concepts for development of new dynamic spatial modelling languages.

Questions related to software technology:

Are the concepts included in dynamic spatial environmental modelling languages better than those of system programming languages, for programming the model?

Unlike system programming languages, which provide entities and functions that represent aspects of the computer, dynamic spatial environmental modelling languages provide the model builder with entities and functions pitched at the level of thinking needed for programming dynamic spatial environmental models. Consequently, dynamic spatial modelling languages are conceptually better for model building than system programming languages. As was shown in chapter 2, dynamic spatial environmental modelling languages have several attractive properties for model building that follow from their concept of providing entities and functions at the level of thinking of environmental researchers. Compared to system programming languages environmental modelling languages are better in their reusability of program code, lack of technical details in the program, short development time, and learnability.

Apart from these advantages, the use of dynamic spatial environmental modelling languages for programming has two disadvantages which directly follow from their concepts. First, the range of models that can be programmed with them will always be restricted, since the fixed set of pre-programmed building blocks provided poses limitations regarding the application of the language, because models may need to be developed that cannot be programmed with the building blocks provided. This can partly be solved by adding a wider range of building blocks to these languages, but the disadvantage will always remain for models with exotic model structures. This conceptual disadvantage can only be solved by providing model builders with the possibility to plug in their own functions, written in another language, mostly a system programming language. This is already done in the PCRaster language. The second disadvantage is the slow run times of models programmed with a dynamic spatial environmental modelling language, relative to these written (and optimised) in a system programming language. This disadvantage will be less significant with faster computers, but when run times are really crucial, it is expected that programming in a system programming language will be more efficient. Although dynamic spatial modelling languages have more disadvantages in addition to these noted above, as will be discussed in the answer to the next research question, most of these disadvantages can be attributed to the relative immatureness of dynamic spatial modelling languages, compared to system programming languages, and not by deficiencies in their concepts.

What are the restrictions of existing dynamic spatial environmental modelling languages for executing the procedural step of programming in the model development cycle?

Apart from the two disadvantages given above, which are implicit to the concepts of dynamic spatial environmental modelling languages, the use of existing dynamic spatial modelling languages comes with restrictions mainly caused by their early stage of development. Most of these are expected to be resolved when dynamic spatial environmental modelling languages evolve towards full-grown programming languages, with concepts for environmental model building which are better than these of system programming languages, as noted above. The most important restrictions of existing dynamic spatial environmental modelling languages for the procedural step of programming are given here (see also Camara, 2002).

As was noted in Chapter 2, existing dynamic spatial environmental modelling languages do not provide sufficient means to prevent the occurrence of programming errors (i.e., bugs) in models. Interactive visualisations embedded in the language allow programmers to check their models while building the model script, but automatic debugging mechanisms are not provided. In addition, it is relatively time consuming to include budget checks in storage-flux based models written in dynamic spatial environmental modelling languages. It can be expected that automatic mechanisms for debugging and checking budgets can be added without changing the framework of existing dynamic spatial environmental modelling languages.

Other restrictions of existing dynamic spatial modelling languages are related to their concept of providing pre-programmed building blocks. As noted in the answer to the previous research question, this will always be a restriction, since only a limited set of models can be built. But major steps forward can be made to minimize this disadvantage by adding new entities and functions. The main restrictions that can be solved in this way are 1) their limitation to two-dimensional modelling, while many environmental models need a three dimensional representation of the environment, 2) their restriction to deterministic modelling, while many environmental models need to be stochastic, 3) the relatively small number of spatial functions provided, with an emphasis on explicit solution of differential equations. These restrictions can be solved by extensions to languages without completely changing their concepts, as is shown in this thesis (see answers to research questions below).

Finally, a restriction of existing field based modelling languages treated in this thesis is that they do not provide functionality for modelling of objects. Adding this functionality will need completely new concepts regarding the design of the computational engine of the language and its syntax.

Questions related to science:

Can we extend dynamic spatial environmental modelling languages with (concepts for) functions for efficient programming of three dimensional models?

Chapter 3 describes a prototype dynamic spatial modelling language capable of dealing with data in three spatial dimensions, using an irregular discretisation in the vertical direction. The language provides a syntax which is as intuitive for environmental

modellers as the syntax of existing dynamic spatial modelling languages, while standard spatial functions are included on three dimensional data. The case study described in Chapter 3, and the application of the concepts for three dimensional modelling in Chapter 7, shows the usefulness of the language for constructing dynamic spatial models simulating erosion and deposition, while it can be expected that the language can also be used in many other fields of environmental modelling.

Although the usefulness of the language can only be demonstrated after it has been tested in a wider range of modelling situations, the positive results noted above, show that it is possible to extend existing dynamic spatial environmental modelling languages with functions for programming with three dimensional environmental data. The problem of large data sets that need to be dealt with when modelling in three dimensions can be resolved by adding optimisation algorithms.

Can we extend dynamic spatial environmental modelling languages with (concepts for) functions for efficient programming of stochastic models in order to calculate error propagation in dynamic spatial models?

The prototype stochastic modelling language and the case studies with it described in Chapter 6, show that it is indeed possible to extend existing dynamic spatial environmental modelling languages with functions for modelling error propagation in dynamic spatial models. A wider application of this type of environmental modelling languages, will only be possible after additional optimisation algorithms to deal with the large data sets involved in stochastic dynamic spatial modelling using Monte Carlo simulation. In addition, exploratory data analysis techniques specially developed for multi dimensional stochastic data are needed for analysing relations in model inputs and outputs.

9.2 Upscaling

As described in Section 1.3.4, upscaling involves scaling methods needed to change the support of field data to appropriate values of inputs and parameters at the support used by the dynamic spatial model. Below, answers are given to the research questions related to the two case studies involving upscaling methods for infiltration. In addition, answers to research questions touching upon the general issue of upscaling are given, inferred from the knowledge and experience gained from these studies.

Questions related to science:

Scale transfer of infiltration was dealt with by upscaling to 1) the support of a catchment (Chapter 5) and, 2) the support of the units in a rainfall-runoff model (Chapter 6). The research questions related to these issues are answered in this order, followed by one research question regarding upscaling in general.

Is it possible to define an upscaling method to scale infiltration measured at a small support (app. 0.04 m^2) to effective values of infiltration for catchments (1-7500 m^2),

which correspond to values derived from measurements at that larger scale, under steady state conditions of rainfall, runoff and infiltration?

In Chapter 5, I showed that the upscaling method is capable of upscaling saturated conductivity derived from measurements at a small support (0.04 m^2) to values of effective saturated conductivity representative for a catchment with a size between a plot and a hillslope (app. 1-75000 m^2), under steady state conditions of rainfall, runoff and infiltration. Under these conditions, the upscaling method predicts an increase in effective saturated conductivity of a catchment with 1) an increase in rain intensity, 2) increasing mean and decreasing variance, and skewness of saturated conductivity distribution within the catchment, 3) decreasing spatial scale of variation in saturated conductivity within the catchment, 4) increasing size of the catchment and, 5) decreasing bifurcation of the drainage pattern in the catchment.

The direction of these relations corresponds with estimates of effective saturated conductivity derived from field measurements in other studies. In addition, it has been shown that the method is capable of upscaling saturated conductivity from the local scale (app. 0.004 m^2) to the plot scale (app. 1 m^2). The estimates of saturated conductivity at the local scale, derived from ring infiltrometer measurements representing the local scale values, can be upscaled to effective values at the plot scale. These effective values at the plot scale correspond with values derived from rainfall simulation experiments at that scale. So, it seems quite plausible that the upscaling method is capable of calculating realistic effective values of saturated conductivity for catchments with the size of a plot (1 m^2).

Although the concepts of the upscaling method are in principle also valid for upscaling saturated conductivity to catchments with the size of a hillslope (7500 m^2), valid under steady state conditions, these scaled values could not be compared with saturated conductivity values derived from field measurements at that scale, since conditions of rainfall, runoff and infiltration on a hillslope with a natural rainstorm are predominantly transient (i.e. not in steady state).

Does an upscaling method, which scales infiltration measured at a small support (app. 0.04 m^2) to effective values of infiltration for catchments (1-7500 m^2), give better results when applied to a dynamic rainfall-runoff model than using the same model without the transfer function?

The upscaling method can be used to derive a relation between rainfall intensity and effective saturated conductivity for the hillslope (7500 m^2), under steady state conditions of rainfall and runoff. This relation was tried in a dynamic rainfall-runoff model simulating discharge from the hillslope. For most rainstorms, the simulated discharge was significantly different from the measured discharge. These unsatisfying results are most probably caused by the transient nature of rainfall and runoff on the hillslope, while the upscaling method used to calculate the effective saturated conductivity assumes steady state conditions. However, effective values for saturated conductivity did at least create runoff, which was not the case when directly using the average derived with the field measurements of infiltration. Moreover, using an approach that varied effective saturated conductivity with rainfall intensity gave better results than using a fixed saturated

conductivity per event, or a value obtained from inverse modelling. This indicates that the upscaling procedure to derive effective saturated conductivity that varies with rainfall intensity does have some potential in transient runoff modelling.

When an upscaling method scaling infiltration measured at a small support (app. 0.04 m^2) is used to derive effective values of infiltration for model units (app. 100 m^2) in a dynamic rainfall-runoff model, does this rainfall-runoff model give better results regarding discharge from a hillslope (app. 7500 m^2) and a catchment (app. 0.4 km^2) than these found when the upscaling method is not used?

The research in Chapter 6 showed that it is possible to define a transfer function calculating scale transfer of saturated conductivity from the local scale (app. 0.04 m^2) to the scale of model units (app. 100 m^2) in a dynamic rainfall-runoff model. The transfer function can be parameterised for all model units in the rainfall-runoff model, using data mostly available in rainfall-runoff modelling studies. When the function is used in the dynamic rainfall-runoff model, effective saturated conductivity values for each model unit are calculated that vary with rainfall intensity and the inflow flux from neighbouring units. The simulations for the two example catchments using the transfer function resulted in a smaller difference between simulated and measured discharge than found when the transfer function was not used.

Compared to the approach when local scale values of saturated conductivity are scaled to effective values of a catchment as a whole (previous two research questions), the approach using effective values for model units seems to be more promising for predicting discharge from catchments, mainly because 1) the approach using effective values for units does not suffer from transient conditions within the spatio-temporal domain for which the effective values are calculated, which may not be the case for the approach using effective values for a catchment as a whole, and 2) results of the dynamic simulations with the effective values for model units are slightly better than these found when effective values for a catchment are used.

Is the existing theory sufficient for solving problems of upscaling related to estimating inputs and parameters in a model, in all fields of dynamic spatial environmental modelling?

As noted in the introduction (Section 1.3.4), general theory for solving scale problems has recently been described in standard text books. Although this provides a theoretical background for estimating inputs and parameters in dynamic spatial environmental models, the sometimes unsatisfying results of the case studies of infiltration have shown that for a specific scaling problem, it can still be very difficult to find an upscaling method that gives satisfactory results. This is mainly caused by difficulties in representing spatio-temporal processes related to the specific input or parameter to be scaled, and not by deficiencies in the general theory of upscaling.

So, the existing general theory for upscaling provides general rules how an upscaling method can be developed. But this theory does not automatically provide appropriate upscaling methods in all situations of modelling, for all inputs and parameters, since the upscaling method needs to be developed for each situation of modelling. For many of

these specific situations, upscaling methods have not been described, or are not yet fully developed.

Questions related to technology:

Is the existing software technology sufficient for solving problems of upscaling related to estimating inputs and parameters of a rainfall-runoff model, and in other upscaling situations?

Standard software specially developed for upscaling does not exist. Instead, a wide range of tools needs to be used. For the studies of upscaling of infiltration (Chapter 5 and 6), the steady state stochastic model representing the process of rainfall and runoff was built with a prototype of the stochastic modelling language described in Chapter 4, providing standard functionality for Monte Carlo simulation. In addition, the Gstat software (Pebesma and Wesseling, 1998) was used for simulation of random fields, and a statistical package was used for analysing results and fitting the transfer function described in Chapter 6. Several additional programs were specifically written to manage data exchange between these packages.

From the case studies of upscaling of infiltration it is obvious that the implementation of an upscaling method with the existing software involves considerable more work than, for example, programming a dynamic spatial model with existing dynamic spatial modelling languages. Although it can be put forward that upscaling is more complicated and more case specific than dynamic spatial model building, it seems that software technology for upscaling is somewhat less developed, compared to the technology for constructing dynamic spatial models.

The development of better, easy to use tools for upscaling is expected to be difficult, since methods of upscaling depend on the kind of attribute to be upscaled, the support for which effective values are needed, and the kind of model in which the values are needed. A closer integration of the above mentioned software tools will be a first step towards the development of a good toolbox for upscaling.

9.3 Inverse modelling

As noted in section 1.3.5, inverse modelling is a means to estimate inputs and parameters of a model by comparison of a set of outputs of the model with measurements of these outputs. The answers to the research questions below are mainly based on Chapter 7, which provided a case study of an inverse modelling procedure for predicting three dimensional sedimentary architecture.

Questions related to science:

Does existing scientific knowledge provide sufficient means to make predictions of three dimensional sedimentary architecture with a dynamic spatial model, conditioned to observations?

The case study shows that prediction of three dimensional sedimentary architecture with a dynamic spatial model is indeed possible, for the case of a hypothetical set of observations, but it was not demonstrated that the approach is applicable to real-world problems, because the model and the hypothetical data set are relatively simple. More research aiming at testing the approach with real-world data sets, needs to be directed to a further improvement of dynamic spatial models simulating the processes of sedimentation and erosion, because existing models do not include all processes which are important at the scale of modelling. This should result in a prediction of sedimentary architecture which is more realistic. Further, these models need to be provided with correct values for the probability distribution of parameters and inputs, which is only possible from an analysis of soft information, such as paleoflow direction, aggradation rates, for each case study under consideration. Finally, future research in this field needs to aim at a further improvement of inverse modelling procedures themselves, and the development of a procedure to select the appropriate inverse modelling algorithm for the situation of modelling under consideration, for instance by using genetic algorithms. It is expected that this will significantly decrease run times compared to those found with the brute force procedure applied in the case study.

Does existing scientific knowledge provide sufficient means to do inverse modelling with dynamic spatial environmental models?

It is difficult to extrapolate the results of the case study of sedimentary architecture modelling to the general issue of inverse modelling, since the results of an inverse modelling procedure will only be satisfactory when the conditions are fulfilled for an environmental model with an appropriate representation of processes, and a correct a priori estimation of (the probability distribution of) inputs and parameters. When these conditions are fulfilled, standard inverse modelling techniques will be sufficient for executing the inverse modelling procedure, in most cases, although run times can be large. Run times can be decreased by improving techniques for inverse modelling, or technological innovation (see below).

Questions related to technology:

Does existing computer technology provide sufficient means to make predictions of three dimensional sedimentary architecture with a dynamic spatial model, conditioned to real-world observations?

The computer technology used in the case study was appropriate for an inverse modelling procedure with a simple model and a small number of observations. With a real world data set, the approach is expected to suffer from long run times, mainly because the larger number of observational data and the need to use a more complex dynamic spatial model, aiming at predictions which are more realistic. As noted in the research question related to science, run times can be decreased by using correct (distributions of) input parameters and an inverse modelling procedure which is suitable for this kind of inverse modelling.

Besides these scientific issues, innovative computer technology is needed for application of the approach to real-world observations. This needs to involve both

improvement of software and hardware. Although dynamic spatial environmental modelling languages are attractive for model building, models built with these languages tend to be slower than these built and optimised with a system programming language, as noted in chapter 2. This can be a problem when a model built with a dynamic spatial modelling language is used in an inverse modelling procedure, since the model needs to be executed iteratively. This problem can be solved by 1) a further improvement of dynamic spatial modelling languages, resulting in shorter run times of models, or 2) programming the dynamic spatial model using a system programming language. The latter needs specialist programmers, able to optimise the program code of a system programming language.

Apart from the software used for inverse modelling, the hardware is an important factor determining run times. In the case study described, inverse modelling runs were done on a standard 200 MHz Linux machine. Considerable decrease of run times is expected when faster machines are used, such as a faster personal computer or a vector machine, or a distributed computer with multiple nodes, allowing parallel processing.

Does existing computer technology provide sufficient means to do inverse modelling with dynamic spatial environmental models, in practice?

As noted above, existing computer technology is appropriate for inverse modelling with dynamic spatial environmental models, only when 1) an inverse modelling problem can be solved without too many iterations, and 2) the model run time is sufficiently small. For more complex (slower) models, and an inverse modelling problem that needs more iterations to be solved, run times of the inverse modelling procedure can be a problem. As noted in the previous research question, this problem can partly be solved by innovation of software and hardware, although these improvements in computer technology will not solve the problem of run times completely. There will always be inverse modelling situations which are not feasible, because they need too much run time.

Another issue is the easy use of inverse modelling procedures in combination with models built with a dynamic spatial environmental modelling language. Currently, the software providing the inverse modelling procedures needs to be linked to the dynamic spatial model, for each case study, which may involve additional programming. An integration of inverse modelling procedures with dynamic spatial modelling languages would make inverse modelling available to a wider group of users.

9.4 Training

As noted in section 1.3.6, training model building involves teaching all steps of the model development cycle (Figure 1) to people with some background in environmental sciences, but without, or with little, background in model building and computer programming.

Does the existing PCRaster environmental modelling language, its associated course material and tools for distance learning, provide an efficient means for teaching dynamic spatial model building in all phases of education, for a wide range of people?

At the first level of training, students are not yet prepared for building models themselves, and dynamic spatial models can be used to explain environmental processes and models, where students evaluate models with a fixed model structure by changing model parameters in a graphical user interface. At this level of teaching, models are built by the tutor, which is relatively easy with PCRaster, since it provides an easy to use modelling language, as noted in the answers to the research questions in Section 9.1. In addition, PCRaster comes with standard software for construction of a graphical user interface, on top of a model script (program), which can be done without too much additional work. So, at this level of teaching, PCRaster provides an efficient means for teaching.

The PCRaster software and course materials are also efficient for teaching at the second level of training. At the second level, students have sufficient background knowledge in environmental processes and computer modelling, to start building models themselves. At this level, students are trained in the phase of programming in the model development cycle, using workbooks teaching them to build a model step-by-step. For teaching programming, PCRaster is very efficient because models can be programmed using building blocks pitched at the level of thinking of most environmental researchers. In addition, PCRaster comes with software manuals and course materials at various levels of programming, providing the possibility for distance learning too.

At the third level of training, students need to be taught in all procedural steps of the model development cycle. This is done in relation with a field study in their own research area. Students need to learn how to identify the model structure on the basis of the aim of their research, data availability and the key processes in their study area. Although model structure identification is mainly done using knowledge of processes important in the specific research, the PCRaster software plays an important role in this model structure identification phase, since it allows students to build and test models themselves, which is not possible when standard user models are used. Teaching the procedural step of estimation of inputs and parameters in a model receives insufficient attention in the PCRaster training material, partly because standard software for these issues is not provided, but also because teaching material is not available. For teaching this step of model building, additional software and course material is needed.

9.5 Central research question

Is the current state of computer technology and science sufficient for executing the steps of programming, upscaling, and inverse modelling in the model development cycle, and for teaching all steps in the model development cycle, with respect to dynamic spatial environmental models simulating continuous fields?

Model building in a specific case study will only result in the optimal model for that case study, when all steps of the model development cycle are provided with optimal scientific knowledge and computer tools, while environmental researchers need to be trained to go through these steps. This thesis evaluated three steps in the model development cycle: programming, upscaling, and inverse modelling, and the issue of training people. The step of programming is mainly related to computer technology. Existing dynamic spatial environmental modelling languages provide tools for programming which are attractive

for environmental researchers, since the concepts of these tools are pitched at the level of thinking of environmental researchers. The extensions to existing dynamic spatial modelling languages with concepts for three dimensional modelling and stochastic modelling described in this thesis, show that limitations of dynamic spatial environmental modelling languages can be resolved by adding new concepts to them. It can be concluded that existing dynamic spatial modelling languages are very attractive for programming models in a number of application fields, while extensions are needed to these languages to make them more generic.

General scientific theories exist for the procedural step of upscaling of field data to the support of a dynamic spatial environmental model. The case studies into infiltration showed that using these general theories and theory in the field of hydrology, it is possible to define an upscaling method for infiltration which gives satisfactory results in dynamic spatial rainfall-runoff models. The success of these case studies does not prove that the scientific issue of upscaling is resolved, or can be resolved, for all situations of modelling. This is mainly because upscaling methods are highly specific for the property of interest , and the situation of modelling. Upscaling can be done with existing software tools, but improvements are needed to make these tools more easily accessible, to a wider group of researchers.

The case study into sedimentary architecture modelling showed that inverse modelling is possible in principle with the existing theory for inverse modelling procedures and the available computer technology. This does not mean that inverse modelling will be successful in all cases, since the quality of the outcome of an inverse modelling procedure depends on the quality of the dynamic spatial environmental model used, which is case study specific, while many inverse modelling procedures will not be feasible because they need too much computation time. The latter issue of run times can partly be dealt with by better hardware and software technology, and an improvement of inverse modelling procedures.

The PCRaster software and training materials provide good means to train people in the procedural steps in the model development cycle of model structure identification and programming. The steps of upscaling and inverse modelling do not come with standard training material, partly because these procedural steps are not provided as standard software in PCRaster.

The general conclusion is that existing dynamic spatial modelling languages provide a good means for performing the procedural step of programming in the model development cycle. The other steps of upscaling and inverse modelling come with general theories, but these steps need to be executed taking care of the specific situation of the case study under consideration. In addition, software which is easier accessible for environmental modellers, and faster hardware, is needed for a more efficient execution of the steps of upscaling and inverse modelling. With regard to training people in model building, sufficient materials are available for training the steps of model structure identification and programming, while more materials are needed for training the steps of upscaling and inverse modelling.

9.6 References

Camara, A.S., 2002. *Environmental Systems: A Multidimensional Approach*. Oxford University Press, Oxford.

Pebesma, E. and C.G. Wesseling, 1998, Gstat: a program for geostatistical modelling, prediction and simulation, *Computers & Geosciences* 24(1), pp. 17-31.

Summary

An environmental model is a representation or imitation of complex natural phenomena that can be discerned by human cognitive processes. This thesis deals with the type of environmental models referred to as dynamic spatial environmental models. The word 'spatial' refers to the geographic domain which they represent, which is the two- or three-dimensional space, while 'dynamic' refers to models simulating changes through time using rules of cause and effect, represented in mathematical equations. Since these equations generally include complex interactions which can only be solved by numerical solution, dynamic spatial models are programmed and run on a computer.

The aim of dynamic spatial environmental model building is to find the optimal representation of environmental processes in the numerical equations (and parameters) of a computer program of the model, for a given case study defined by the aim of modelling, the properties of the study site, the field data present, the software en hardware technology available to construct the model, and the researchers involved. Since most of these factors will be different for each case study, a new (or modified) model should be made for each case study, by executing the procedural steps of the model development cycle (Figure 1) until the optimal model has been found.

This thesis evaluates whether existing technology and/or science provide sufficient means to deal with four issues related to the model development cycle (Figure 1). The first issue is concerned with programming the model. Environmental modelling languages exist which are programming languages specifically meant for building dynamic spatial environmental models. It is shown in the thesis that concepts of these languages are better than these of system programming languages when used for programming an dynamic spatial environmental model. However, the disadvantage of existing environmental modelling languages is their restricted set of models that can be built with them. To overcome this disadvantage, concepts, a prototype language, and case studies, are described for new environmental modelling languages supporting construction of three dimensional and stochastic dynamic models for calculating error propagation in environmental models. Construction of these kind of models was so far not possible with environmental modelling languages.

The second issue which evaluated is upscaling methods, which are procedures describing how to derive model inputs and parameters, representative for the support (resolution) of the dynamic spatial environmental model, from field data collected at a smaller support. This is explored in two case studies involving upscaling of local measurements of infiltration to values of infiltration that can be used in dynamic spatial models simulating runoff in a catchment during a rainstorm. In the first case study, representative (effective) values are calculated for a hillslope as a whole, while in the second case study, representative values are calculated for a support corresponding to the model units (grid cells) of a rainfall-runoff model. It is shown that the proposed upscaling methods can be applied in rainfall-runoff models, although problems caused by the transient character of the rainfall-runoff processes, and/or data availability remain to be solved. From the experiences of this case study, it is concluded that upscaling remains a complicated issue, and it will be difficult to develop standard software which makes

upscaling possible for a wider group of researchers, in a wide range of environmental models.

The third issue evaluated is the method of inverse modelling (or inverse estimation) for estimating inputs and parameters of a dynamic spatial model by minimizing the difference between the simulated and measured values of one or a set of outputs of a dynamic spatial environmental model. This is studied in a case study aiming at the three dimensional interpolation of sediment type, using a dynamic spatial model simulating erosion and deposition by rivers, on time scales of thousands of years. It is shown that with a brute force inverse modelling procedure, the sediment type between hypothetical well locations can be predicted with the model, in three dimensions, while at the well locations, the simulated sediment type corresponds to the observed sediment type. It is concluded that the method applied for three dimensional interpolation of sedimentary deposits works in principle, while practical application with a real world data set needs to involve a further decrease of the computer run time of the dynamic spatial model used, and the use of a more advanced inverse modelling procedure.

The fourth issue evaluated is that of training people in executing all steps of the model development cycle. It is shown that existing environmental modelling languages are an efficient tool for training model building in all phases of learning, although progress need to be made regarding software and training material for teaching upscaling and inverse modelling.

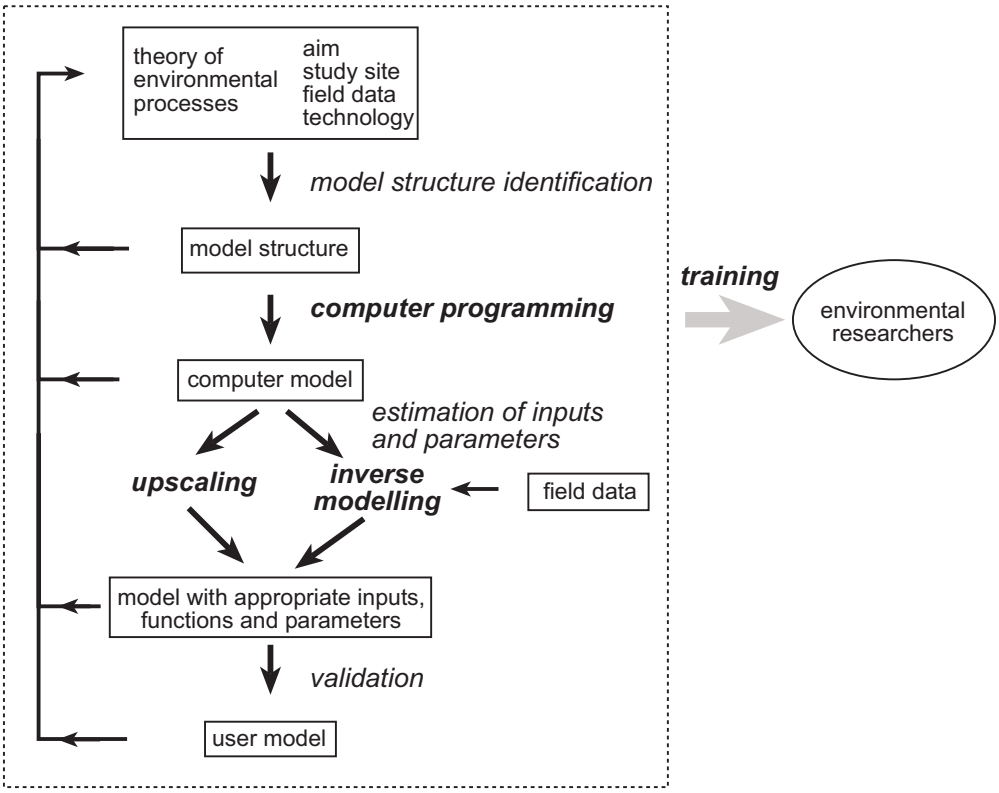


Figure 1. Model development cycle for building dynamic spatial environmental models. Issues of programming, upscaling, inverse modelling and training (in bold type) are these dealt with in the thesis.

Het maken van dynamische ruimtelijke landschapsmodellen

Samenvatting

Een landschapsmodel is een representatie of imitatie van complexe natuurlijke fenomenen die kunnen worden onderscheiden door menselijke cognitieve processen. Dit proefschrift behandelt het type landschapsmodellen dat bekend staat als dynamische ruimtelijke modellen (*dynamic spatial environmental models*). Het woord 'ruimtelijk' verwijst hierbij naar de twee- of driedimensionale ruimte waarbinnen de processen zich afspelen. Het woord 'dynamisch' verwijst naar modellen die veranderingen in de tijd simuleren. Deze veranderingen worden gemodelleerd met gebruik van regels van oorzaak en gevolg, welke worden weergegeven in wiskundige vergelijkingen. Doordat de verbanden die weergegeven worden door deze vergelijkingen complex kunnen zijn, moeten de vergelijkingen worden opgelost met behulp van numerieke oplossingstechnieken, beschreven in een computerprogramma dat alle modelvergelijkingen bevat.

Het doel van de constructie van dynamische ruimtelijke landschapsmodellen is het vinden van de optimale weergave van landschappelijke processen in numerieke vergelijkingen (en parameters) van een computer programma van het model, voor een gegeven onderzoek. Hierbij is het gegeven onderzoek gedefinieerd door het doel van de modelleerprocedure, de eigenschappen van het onderzoeksgebied, de veldgegevens die beschikbaar zijn, de beschikbare software en hardware technologie voor de ontwikkeling en het draaien van het model, en de onderzoekers die het model maken. Aangezien deze factoren specifiek zijn voor een bepaald onderzoek, is het in de meeste gevallen nodig om een nieuw, of aangepast, model te maken voor ieder onderzoek. De ontwikkeling van een nieuw model dient te gebeuren door middel van het doorlopen van de procedurele stappen van de modelbouwcyclus (Figuur 1), waarbij verschillende modelrepresentaties van processen in het landschap kunnen worden getest, totdat het optimale model is gevonden.

In dit proefschrift wordt geëvalueerd of de huidige technologie en wetenschap voldoende kennis en middelen ter beschikking heeft om de modelbouwcyclus bevredigend te doorlopen. Hierbij wordt gekeken naar vier onderdelen van de cyclus. Het eerste onderdeel waarnaar wordt gekeken is het programmeren van het model. Hiervoor bestaan landschapsmodelleertalen (*environmental modelling languages*) die specifiek zijn ontwikkeld voor het programmeren van dynamische ruimtelijke modellen. Het wordt in dit proefschrift aangetoond dat voor het programmeren van een dynamisch ruimtelijk model dergelijke landschapsmodelleertalen conceptueel beter zijn dan systeemprogrammeertalen. Het nadeel van bestaande landschapsmodelleertalen is echter dat er slechts een beperkte set van modellen mee gemaakt kunnen worden, doordat nog onvoldoende functionaliteit beschikbaar is. Om dit nadeel voor een deel op te lossen, worden in dit proefschrift concepten voor landschapsmodelleertalen beschreven die de ontwikkeling mogelijk maken van 1) driedimensionale en 2) stochastische dynamische modellen. Dergelijke modellen konden tot nu toe niet met de bestaande landschapsmodelleertalen worden gemaakt. Tevens worden prototypen beschreven van modelleertalen die ontwikkeld zijn volgens deze concepten.

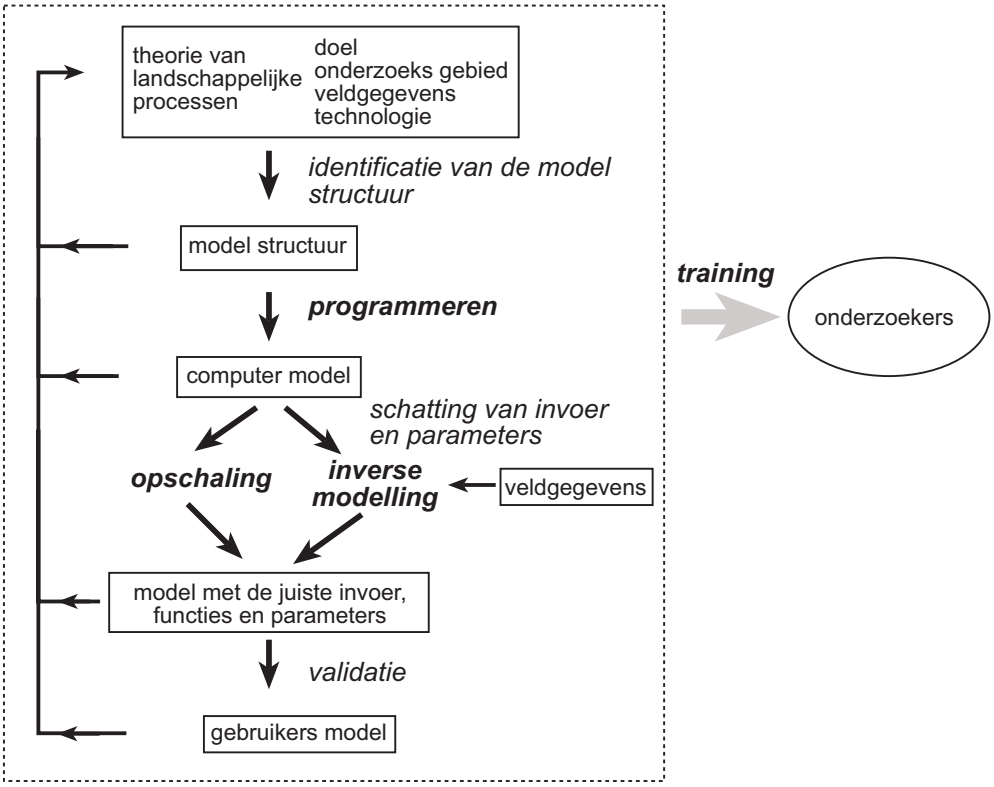


Figure 1. Modelbouwcyclus voor de ontwikkeling van een dynamisch ruimtelijk landschapsmodel. De stappen programmeren, opschaling, *inverse modelling* en training (vet letter type) worden behandeld in het proefschrift.

Het tweede onderdeel dat wordt geëvalueerd is opschaling. Een opschalingsmethode beschrijft hoe veldgegevens die verzameld zijn met een kleine *support* kunnen worden omgezet in waarden voor de invoer of parameters van een model die representatief zijn voor de grotere *support* van het model. Dit onderwerp wordt behandeld in twee onderzoeken naar het opschalen van puntmetingen van infiltratie naar waarden representatief voor de *support* van een dynamisch ruimtelijk model dat afvoer in een stroomgebied simuleert tijdens een regenbui. In het eerste onderzoek worden waarden berekend die representatief zijn voor een gebied ter grootte van een helling. Het tweede onderzoek behandelt de berekening van waarden van infiltratie die representatief zijn voor een *support* ter grootte van modeleenheden (gridcellen) van een regen afvoer model. In de twee onderzoeken wordt aangetoond dat de voorgestelde opschalingsmethoden kunnen worden gebruikt in regen afvoer modellen. Dit gaat echter gepaard met enkele problemen die nog zullen moeten worden opgelost. Deze problemen zijn gerelateerd aan het feit dat tijdens een regenbui geen lange perioden optreden waarbij sprake is van een stationaire toestand van alle fluxen. Daarnaast is het een probleem dat onvoldoende veldgegevens beschikbaar zijn. Uit de resultaten van de twee onderzoeken kan worden geconcludeerd dat opschaling een complex probleem is. Het is niet te verwachten dat het eenvoudig zal zijn om standaard software voor opschaling te maken die opschaling toegankelijk maakt voor een bredere groep van onderzoekers, in een breed scala van landschapsmodellen.

Het derde onderdeel van de modelbouwcyclus dat wordt behandeld is de methode die bekend staat als *inverse modelling* (of *inverse estimation*) voor het schatten van de invoer en de parameters van een dynamisch ruimtelijk model. Een *inverse modelling* procedure schat de waarden van de invoer en de parameters door deze zodanig aan te passen totdat het verschil tussen de gesimuleerde en de gemeten waarden van één of een set van uitvoeren van het dynamisch ruimtelijk model is geminimaliseerd. Dit onderwerp wordt behandeld in een voorbeeldstudie die tot doelstelling heeft om het type sediment in een rivierafzetting te interpoleren tussen hypothetische boorpunten, in drie dimensies. Dit wordt gedaan door middel van een dynamisch ruimtelijk model dat erosie en depositie door rivieren simuleert, op een tijdschaal van duizenden jaren. Het wordt aangetoond dat met een eenvoudige *inverse modelling* procedure het sedimenttype tussen hypothetische boorpunten kan worden voorspeld, waarbij op de locatie van de boorpunten zelf het sedimenttype wordt voorspeld dat in de boringen is gevonden. De conclusie is dat de toegepaste methode voor driedimensionale interpolatie van sedimentaire afzettingen in principe werkt. De praktische toepassing van de methode met gebruik van werkelijke gegevens vereist verbeteringen wat betreft de rekentijd van het gebruikte dynamische model, en verbeteringen in de *inverse modelling* procedure zelf.

Het vierde onderwerp dat wordt geëvalueerd is het trainen van mensen in het uitvoeren van alle stappen in de modelbouwcyclus. Het wordt aangetoond dat bestaande landschapsmodelleertalen een efficiënt gereedschap vormen voor onderwijs in het maken van modellen in alle fasen van het leerproces, hoewel vooruitgang nodig is wat betreft de software en het cursusmateriaal voor onderwijs in opschalen en *inverse modelling*.

Acknowledgements

Many thanks to Peter Burrough for having faith in me, for his support through the years, and for supervising the research in this thesis. Marc Bierkens is gratefully acknowledged for his help, advice and for many inspiring discussions. I would like to thank Willem van Deursen for his understanding, his enthusiasm and many important comments. The work described in his thesis on Geographical Information Systems and Dynamic Models, written in 1995, was an important inspiration for this research.

One of my main joys in science is being in a research group developing dynamic spatial modelling tools. Many thanks to Cees Wesseling for working with me and to Willem van Deursen for his support. Peter Burrough is thanked for all the effort he puts into our research. Edzer Pebesma, Victor Jetten, Marcel van der Perk and many other colleagues are thanked for working on and supporting software development. I would like to thank Kor de Jong for being a great colleague, and for providing important inputs to this thesis. Kor developed all PCRaster visualisation routines used here, and the prototype modelling language described in chapter 3 and 4.

I am very grateful to other people who played a crucial role in the work described in this thesis. Among them are Lorna Booth, Simone van Dijck, John Bridge, Torbjörn Törnqvist, Edzer Pebesma, Marlous van der Meer, Raymond Sluiter, Rodrigo Oliveira, João Ribeiro da Costa, and Arko Lucieer.

My colleagues at the institute of Physical Geography are acknowledged for inspiring my research in many different ways. I would like to mention Theo van Asch, Marcel van der Perk, Anja de Wit, Rens van Beek, Karin Pfeffer, Steven de Jong, Thom Bogaard, Wladimir Bleuten, Martin Hendriks, Henk van Steijn, Rob Kromwijk en Rien Rabbers.

My greatest thanks to my mother, Ferd, Karin, Berend, Susanne, Hans, Mariken, Rob, Annemiek, Joanita, Alex, Ira, Kitty, Hans, Angelika, Naomi, Bart, Karin, Lorna, Dave, Luciano, Federica, Alireza, Øystein, Hajo, Tim, Ana, for their support and care.

Curriculum Vitae

Derek Karssenberg was born on 25th November 1968 in Utrecht, the Netherlands. In 1988 he passed his final exams at the Baudartius College secondary school, Zutphen. From 1988 till 1994 he studied Physical Geography at Utrecht University, and in 1994, joined the PCRaster research group at the Department of Physical Geography, Utrecht University. As a member of this group, he wrote course materials and the manual for the PCRaster software which was released in 1996 and organised short courses in environmental modelling. In 1997 he obtained a position at the same department as a lecturer in environmental modelling and hydrology, teaching dynamic spatial environmental modelling, hydrology, land degradation, (geo)statistics, and Geographical Information Systems. In this period, he continued working in the PCRaster team at Utrecht University, focussing his research on catchment modelling, alluvial architecture modelling, development of software for building environmental models, and development of teaching methods for training environmental modellers over the internet.

Publications

Peer-reviewed publications

Brama, P.A.J., D. Karssenberg, A. Barneveld & P.R. Weeren (2001), Contact areas and pressure distribution on the proximal articular surface of the proximal phalanx under sagittal plane loading. Equine Veterinary Journal 33, pp. 26-32.

Brama, P.A.J., J.M. Tekoppelle, R.A. Bank, D. Karssenberg, A. Barneveld & P.R. van Weeren (2000), Topographical mapping of biochemical properties of articular cartilage in the equine fetlock joint. Equine Veterinary Journal 32, pp. 19-26.

Karssenberg, D. (2002), The value of environmental modelling languages for building distributed hydrological models. Hydrological Processes, in press.

Karssenberg, D., P.A. Burrough, R. Sluiter & K. de Jong (2001), The PCRaster software and course materials for teaching numerical modelling in the environmental sciences. Transactions in GIS 5, pp. 99-110.

Karssenberg, D., T. Törnqvist & J.S. Bridge (2001), Conditioning a process-based model of sedimentary architecture to well data. Journal of Sedimentary Research 71, pp. 868-879.

Wesseling, C.G., D. Karssenberg, W.P.A. van Deursen & P.A. Burrough (1996), Integrating dynamic environmental models in GIS: the development of a Dynamic Modelling language. Transactions in GIS 1, pp. 40-48.

Course materials

Karssenberg, D., E.J. Pebesma, M. van der Meer & P.A Burrough (2000), PCRaster en Gstat distance learning course material: An introduction to interpolation and geostatistical Modelling. Appr. 30 pp. Info at http://www.geog.uu.nl/pcraster

Karssenberg, D., W.P.A. van Deursen, C.G. Wesseling, M. van der Meer, K. de Jong & P.A. Burrough (2000), PCRaster distance learning course material: Map algebra and environmental modelling. Appr. 50 pp. Info at http://www.geog.uu.nl/pcraster

Karssenberg, D., W.P.A. van Deursen, C.G. Wesseling, M. van der Meer, P.A. Burrough & K. de Jong (2000), PCRaster distance learning course material: An introduction to dynamic modelling. Appr. 50 pp. Info at http://www.geog.uu.nl/pcraster

Karssenberg, D., C.G. Wesseling & W.P.A. van Deursen (1996), PCRaster version 2, manual. Appr. 380 pp. Available at http://www.geog.uu.nl/pcraster

Sluiter, R., P.A. Burrough, D. Karssenberg, A. Lucieer, C.G. Wesseling, K. de Jong, T.W.J. van Asch, H. van Steijn, M. Cadee, R.J. de Boer, W.P.A. van Deursen & M.R. Hendriks (2000), Virtual Landscapes, interactive models for learning about geographical change. Appr. 50 pp. Available at: http://www.virtualland.geog.uu.nl

Reports

Vandenhove, H., F. Goor, S. Timofeyev, V. Ageyets, V. Averin, O. Voitsekhovitch, N. Victorova, B. Sorochinsky, M. van der Perk & D. Karssenberg (1999), PHYTOR - Evaluation of willow plantations for the phytorehabilitation of contaminated arable lands and floodplain

areas, EC Contract No. ERB IC15 CT98 0213; Intermediary Reprt #1. Mol: SCK-CEN, 76 pp.

Conference publications

Burrough, P.A., C. Wesseling, R. Sluiter, D. Karssenberg, K. de Jong & M. van der Meer (2000), Virtual landscapes, interactive models for learning about geographical change. 4th International Conference on Environmental Modelling, Problems, Prospects and Research Needs. Banff, Alberta, Canada, september 2-8.

De Jong, K., P.A. Burrough, D. Karssenberg & E.J. Pebesma (2000), An environmental modeling language for model construction in the temporal, 3D spatial and stochastic dimension: prototype. Proceedings of the 4th International Conference on Integrating GIS and Environmental Modeling, September 2-8, Banff, Canada. http://www.colorado.edu/research/cires/banff

De Jong, K., C.G. Wesseling & D. Karssenberg (2002), Improving the model development cycle by automatic configuration of modelling tools. Proceedings iEMSs 2002, Integrated Assessment and Decision Support, 24-27 June, 2002, Lugano, Switzerland.

Karssenberg, D. (2002), Geographical Information Systems and Hydrological Modelling. Keynote, conference proceedings EuroConference "Link Geo and Water Research"", Febraury 7th-9th, 2002, Genova, Italy.

Karssenberg, D., J.S. Bridge & K. de Jong (2001), Modeling cycles of fluvial aggradation and degradation due to base-level change using a revised process-based, 3D alluvial stratigraphy model. Proceedings Fluvial Sedimentology 2001, 7th International Conference on Fluvial Sedimentology, Lincoln, Nebraska, August 6-10, 2001.

Karssenberg, D., T. Törnqvist & J.S. Bridge (2001), Conditioning a Process-Based Model of Sedimentary Architecture to Well Data. Proceedings Fluvial Sedimentology 2001, 7th International Conference on Fluvial Sedimentology, Lincoln, Nebraska, August 6-10, 2001.

Karssenberg, D. & P.A. Burrough (2000), Teaching numerical modelling in the environmental sciences. Keynote, proceedings of the 4th International Conference on Integrating GIS and Environmental Modeling, September 2-8, Banff, Canada. http://www.colorado.edu/research/cires/banff

Karssenberg D., K. de Jong & P.A. Burrough (2000), A prototype computer language for environmental modeling in the temporal, 3D spatial and stochastic dimension. Proceedings of the 4th International Conference on Integrating GIS and Environmental Modeling, September 2-8, Banff, Canada. http://www.colorado.edu/research/cires/banff

Pebesma, E.J, D. Karssenberg & K. de Jong (2000) The stochastic dimension in a dynamic GIS. In: J.G. Bethlehem, P.G.M. van der Heijden, editors. Compstat 2000, Proceedings in Computational Statistics. 14th Conference of the International Association for Statistical Computing, 21-25 August. Physica-Verlag, Heidelberg, 379-384

Perk, M. van der, V.G. Jetten, D. Karssenberg, Q. He, D.E. Walling, G.V. Laptev, O.V. Voitsekhovitch, A.A. Svetlichnyi, O. Slavik, V.G. Linnik, E.M. Korobova, S. Kivva & M. Zheleznyak (2000), Assessment of spatial redistribution of Chernobyl-derived radiocaesium within catchments using GIS-embedded models. In: The Role of Erosion and Sediment Transport in Nutrient and Contaminant Transfer(ed. by M. Stone) (Proc. Waterloo Symp., July 2000). IAHS Publ. no. 263, pp. 277-284.

Pfeffer, K., D. Karssenberg, T.W. van Asch & P.A. Burrough (2002), Application of spatio-temporal environmental modelling and multi-criteria analysis to optimise the planning of

socio-economic infrastructure. Proceedings iEMSs 2002, Integrated Assessment and Decision Support, 24-27 June, 2002, Lugano, Switzerland.
Rebel, K.T., S.J. Riha, K. Karssenberg & D.R. Hitchcock (2002), A Spatial Explicit Watershed Model of Water and Tritium Fluxes in the Vadose Zone. Proceedings American Geophysical Union 2002 Spring Meeting, 28-31 May 2002, Washington.
Sluiter, R., H. van Steijn, D. Karssenberg, P.A. Burrough, C. Wesseling & K. de Jong (2000), Interactive computer models for teaching dynamic geomorphological processes. An example of web-based distance learning. 4th International Conference on Environmental Modelling, Problems, Prospects and Research Needs. Banff, Alberta, Canada, september 2-8
Van Deursen, W., C. Wesseling & D. Karssenberg (2000), How do we gain control over GIS technology? Proceedings of the 4th International Conference on Integrating GIS and Environmental Modeling, September 2-8, Banff, Canada. http://www.colorado.edu/research/cires/banff
Pebesma E.J., D. Karssenberg & K. de Jong (2001), The stochastic dimension in a dynamic GIS. Proceedings of Compstat 2000, 14th Conference of the International Association for Statistical Computing, 21-25 August, Utrecht, the Netherlands.
Wesseling, C.G., W.P.A. van Deursen, D. Karssenberg & P.A. Burrough (1996), Experience and applications of a GIS based dynamic modelling language. Proceedings Second Joint European Conference & Exhibition on Geographical Information, Barcelona, Spain, 1996.

Chapters in books

Van Dijck, S.J.E., D. Karssenberg (2000), Surface runoff. Chapter in S.J.E. van Dijck (2000), Effects of agricultural land use on surface runoff and erosion in a Mediterranean area. Utrecht: Koninklijk Nederlands Aardrijkskundig Genootschap/Faculteit Ruimtelijke Wetenschappen, Universiteit Utrecht.