FINNISH METEOROLOGICAL INSTITUTE
CONTRIBUTIONS

No. 98

DEVELOPMENT, VALIDATION AND APPLICATION OF NUMERICAL SPACE ENVIRONMENT MODELS

Ilja Honkonen

Department of Physics
Faculty of Science
University of Helsinki
Helsinki, Finland

ACADEMIC DISSERTATION in physics

To be presented, with the permission of the Faculty of Science of the University of Helsinki, for public criticism in auditorium D101 at Physicum in Kumpula campus (Gustaf Hällströmin katu 2a) on 4$^{\text{th}}$ of October 2013 at noon (12 o'clock).

Finnish Meteorological Institute
Helsinki, 2013

ISBN 978-951-697-793-8 (paperback)
ISBN 978-951-697-794-5 (pdf)
ISSN 0782-6117

Unigrafia Oy Yliopistopaino
Helsinki, 2013

FINNISH METEOROLOGICAL INSTITUTE

Series title, number and report code of publication
Contributions 98, FMI-CONT-98

Published by Finnish Meteorological Institute
(Erik Palménin aukio 1) , P.O. Box 503
FIN-00101 Helsinki, Finland

Publication month
October 2013

Author(s)

Ilja Honkonen

Name of project

Commissioned by

Title
Development, validation and application of numerical space environment models

Abstract

Currently the majority of space-based assets are located inside the Earth's magnetosphere where they must endure the effects of the near-Earth space environment, i.e. space weather, which is driven by the supersonic flow of plasma from the Sun. Space weather refers to the day-to-day changes in the temperature, magnetic field and other parameters of the near-Earth space, similarly to ordinary weather which refers to changes in the atmosphere above ground level. Space weather can also cause adverse effects on the ground, for example, by inducing large direct currents in power transmission systems.

The performance of computers has been growing exponentially for many decades and as a result the importance of numerical modeling in science has also increased rapidly. Numerical modeling is especially important in space plasma physics because there are no in-situ observations of space plasmas outside of the heliosphere and it is not feasible to study all aspects of space plasmas in a terrestrial laboratory. With the increasing number of computational cores in supercomputers, the parallel performance of numerical models on distributed memory hardware is also becoming crucial.

This thesis consists of an introduction, four peer reviewed articles and describes the process of developing numerical space environment/weather models and the use of such models to study the near-Earth space. A complete model development chain is presented starting from initial planning and design to distributed memory parallelization and optimization, and finally testing, verification and validation of numerical models. A grid library that provides good parallel scalability on distributed memory hardware and several novel features, the distributed cartesian cell-refinable grid (DCCRG), is designed and developed. DCCRG is presently used in two numerical space weather models being developed at the Finnish Meteorological Institute. The first global magnetospheric test particle simulation based on the Vlasov description of plasma is carried out using the Vlasiator model. The test shows that the Vlasov equation for plasma in six-dimensionsional phase space is solved correctly by Vlasiator, that results are obtained beyond those of the magnetohydrodynamic (MHD) description of plasma and that global magnetospheric simulations using a hybrid-Vlasov model are feasible on current hardware. For the first time four global magnetospheric models using the MHD description of plasma (BATS-R-US, GUMICS, OpenGGCM, LFM) are run with identical solar wind input and the results compared to observations in the ionosphere and outer magnetosphere. Based on the results of the global magnetospheric MHD model GUMICS a hypothesis is formulated for a new mechanism of plasmoid formation in the Earth's magnetotail.

Publishing unit
Earth observation

Classification (UDK)
004.415.2, 004.415.5, 004.421.032.24,
519.17, 532.5:519.6, 537.8, 551.510.535

Keywords
Space weather model development,
verification, validation

ISSN and series title
0782-6117 Finnish Meteorological Institute Contributions

ISBN
978-951-697-793-8 (paperback), 978-951-697-794-5 (pdf)

Language
English

Sold by
Finnish Meteorological Institute / Library
P.O.Box 503
FIN-00101 Helsinki, Finland

Pages
148

Price

Note

ILMATIETEEN LAITOS

Julkaisun sarja, numero ja raporttikoodi
Contributions 98, FMI-CONT-98

Julkaisija Ilmatieteen laitos
(Erik Palménin aukio 1), PL 503
00101 Helsinki

Julkaisuaika
Lokakuu 2013

Tekijä(t)

Ilja Honkonen

Projektin nimi

Toimeksiantaja

Nimeke
Laskennallisten avaruussäämallien kehittäminen, validointi ja käyttö

Tiivistelmä

Avaruuteen lähetetyistä laitteista suurin osa sijaitsee Maan magnetosfäärissä, missä ne altistuvat avaruussäälle. Avaruussäällä tarkoitetaan Maan lähiavaruuden läpötilan, magneettikentän ja muiden ominaisuuksien päivittäistä vaihtelua auringosta jatkuvasti virtaavan plasman - aurinkotuulen - vuoksi. Avaruussäällä voi olla haitallisia vaikutuksia myön Maan pinnalla, esimerkkinä sähkönsiirtoverkkoihin indusoituvat suuret tasavirrat.

Tietokoneiden laskentateho on kasvanut eksponentiaalisesti jo vuosikymmenien ajan, minkä seurauksena myös laskennallisen mallinnuksen merkitys tieteelle on kasvanut huomattavasti. Laskennallinen mallintaminen on erityisen tärkeää avaruusplasmafysiikassa, sillä aurinkokunnan ulkopuolelta ei ole suoria mittauksia, eikä kaikkia avaruusplasman ominaisuuksia voida tutkia maanpäällisissä laboratorioissa. Supertietokoneiden laskentaytimien määrän kasvaessa myös laskennallisten mallien rinnakkaisesta suorituskyvystä on tullut ratkaisevan tärkeää.

Väitöskirja koostuu johdannosta ja neljästä vertaisarvioidusta julkaisusta joissa kuvataan laskennallisten avaruussäämallien kehittämistä ja käyttöä Maan lähiavaruuden tutkimiseen. Avaruussäämallien kaikki kehittämisaskeleet käydään läpi alkaen alustavasta suunnittelusta ja toteutuksesta, jaetun muistin rinnakkaistuksesta ja laskentanopeuden optimoinnista aina testaukseen ja validointiin asti. Väitöskirjan yhteydessä on suunniteltu ja toteutettu rinnakkainen hila jota käytetään tällä hetkellä kahdessa Ilmatieteen laitoksella kehitettävässä avaruussäämallissa. Näistä toisen, Vlasovin yhtälöllä plasmaa mallintavan Vlasiatorin, täyden kuusiulotteisen faasiavaruuden (kolme paikka- ja kolme nopeusulottuvuutta) käsittävällä magnetosfääritestillä on osoitettu mallin toimivuus ja soveltuvuus Maapallon koko magnetosfäärin mallintamiseen nykyisillä supertietokoneilla. Ensimmäistä kertaa on myös vertailtu neljän eri avaruussäämallin (BATS-R-US, GUMICS, OpenGGCM, LFM) tuottamia ennusteita Maan lähiavaruudesta käyttäen samaa aurinkotuulisyötettä.

Julkaisijayksikkö
Uudet havaintomenetelmät

Luokitus (UDK)
004.415.2, 004.415.5, 004.421.032.24,
519.17, 532.5:519.6, 537.8, 551.510.535

Asiasanat
Avaruussäämalli, kehittäminen, validointi

ISSN ja avainnimike
0782-6117 Finnish Meteorological Institute Constributions

ISBN
978-951-697-793-8 (painettu), 978-951-697-794-5 (pdf)

Kieli
Englanti

Myynti
Ilmatieteen laitos / Kirjasto
PL 503
00101 Helsinki

Sivumäärä
148

Hinta

Lisätietoja

Preface

Five years ago my knowledge of space plasma physics was comparable to that of an average physics graduate, that is, I had only taken one course in electrodynamics and had not heard of, for example, magnetic reconnection. The most complex numerical model I had written solved either the heat transfer problem in two dimensions or the motion of point masses under the effect of gravity, implemented using a very inefficient algorithm. During the past five years I have been able to work on cutting edge problems related to fundamental questions in both physics and mathematics, used some of the largest supercomputers in Europe and USA, spent over one million CPU core hours, played Conway's Game of Life using 64008 MPI processes, worked with and learned from the leading experts in space physics. I can only hope to learn as much during the next five years.

First and foremost I'm grateful to Minna Palmroth for the opportunity to write this thesis, to work in one of Europe's leading groups in space plasma modeling and for getting me started on my career in scientific publishing. Pekka Janhunen's expert guidance on numerical modeling in general and GUMICS in particular was also essential. Comments from the pre examiners Ralf Kissmann and Gábor Tóth, totaling over 4000 words, improved this thesis significantly. I thank my coauthors for invaluable advice in creating this thesis' articles, the QuESpace team and past and present roommates for an excellent working environment, and all colleagues at the Finnish Meteorological Institute, University of Helsinki, NASA, St. Petersburg State University and elsewhere for making the past five years awesome. And, of course, I'm grateful to my parents for helping out with everything in every way.

The research in this thesis was carried out at the Finnish Meteorological Institute and the funding (including traveling) was provided by the following parties: Project 218165 of the Academy of Finland, project 200141 (QuESpace) of the European Research Council, projects 260330 (EURISGIC) and 263325 (ECLAT) of the European Community's seventh framework programme, University of Michigan Center for Space Environment Modeling, NASA Community Coordinated Modeling Center, the Aaltonen, Ehrnrooth and Väisälä foundations, the chancellor of University of Helsinki and the European-US Summer School on HPC Challenges in Computational Sciences.

Ilja Honkonen
Helsinki, 9.9.2013

Contents

Research articles and the author's contributions

This thesis consists of an introduction and four articles, which have not been included in prior theses and are published in peer-reviewed journals, referred to as Articles I, II, III and IV in the text:

Article I (Honkonen et al., 2013a)

I. Honkonen, S. von Alfthan, A. Sandroos, P. Janhunen, M. Palmroth. Parallel grid library for rapid and flexible simulation development. *Computer Physics Communications*, 184(4):1297-1309, 2013. ISSN 0010-4655. doi: 10.1016/j.cpc.2012.12.017.

The author's contribution: Conceived the general implementation ideas (Section 3 of the article), developed the code at https://gitorious.org/dccrg, wrote the tests with the exception of the Roe MHD solver used in them which was borrowed from GUMICS-4, carried out the tests, created the figures, wrote most of the manuscript.

Article II (Palmroth et al., 2013)

M. Palmroth, I. Honkonen, A. Sandroos, Y. Kempf, S. von Alfthan, D. Pokhotelov. Preliminary testing of global hybrid-Vlasov simulation: Magnetosheath and cusps under northward interplanetary magnetic field. *Journal of Atmospheric and Solar-Terrestrial Physics*, 99(0):41-46, 2013. ISSN 1364-6826. doi: 10.1016/j.jastp.2012.09.013.

The author's contribution: Modified the computational hybrid-Vlasov model Vlasiator to allow a global test particle simulation (added necessary boundary conditions, added support for using time-dependent electric and magnetic fields from GUMICS results) and carried out the simulation, created Figure 5, participated in the discussion, planning and writing of the manuscript.

Article III (Honkonen et al., 2013b)

I. Honkonen, L. Rastätter, A. Grocott, A. Pulkkinen, M. Palmroth, J. Raeder, A.J. Ridley, and M. Wiltberger. On the performance of global magnetohydrodynamic models in the Earth's magnetosphere. *Space Weather*, 11(5):313-326, 2013. ISSN 1542-7390. doi: 10.1002/swe.20055.

The author's contribution: Carried out the simulations through the Community Coordinated Modeling Center (CCMC), processed and analyzed the simulation results and observational data, created all figures except Figure 8, wrote almost all of the manuscript.

Article IV (Honkonen et al., 2011)

I. Honkonen, M. Palmroth, T.I. Pulkkinen, P. Janhunen, A. Aikio. On large plasmoid formation in a global magnetohydrodynamic simulation. *Annales Geophysicae*, 29(1):167-179, 2011. doi: 10.5194/angeo-29-167-2011.

The author's contribution: Processed simulation data to demonstrate plasmoid formation, created all figures (first one was adapted from Wikipedia), processed observational data and wrote a large part of the manuscript.

The author has also contributed to the following articles which are referenced in the text but are not part of this thesis: Aikio et al. [2013], von Alfthan et al. [2013a], von Alfthan et al. [2013b], Gordeev et al. [2013], Janhunen et al. [2012], Palmroth et al. [2012], Sandroos et al. [2013].

Terminology

> You can know the name of a bird in all the languages of the world, but when you're finished, you'll know absolutely nothing whatever about the bird...
>
> Richard Feynman

1d, 2d, ... - One, two, ... -dimensional or one, two, ... dimensions
AMR - adaptive mesh refinement
au - astronomical unit (mean Earth-Sun distance, 149 597 870 700 m [IAU Commission, 2012]
cell - smallest or elementary unit of volume in a discretized simulation of a physical volume
CPU - central processing unit
EM - electromagnetism, electromagnetic
eV - electron volt, about $1.6 \cdot 10^{-19}$ J, also about $14[km/s] \times \sqrt{E[\mathrm{eV}]/m[\mathrm{u}]}$ for non-relativistic particles
FDM, FEM, FVM - finite difference, element, volume methods
GoL - Conway's Game of Life
GUMICS - a global MHD model for near-Earth space
kinetic - method which resolves kinetic effects in addition to fluid effects (e.g. PIC)
light year - distance traveled by light in one year in a vacuum, about 63 000 au or 10^{16} m
MHD - magnetohydrodynamic(s)
MPI - message passing interface standard and its software implementations
OpenMP - application programming interface for shared memory parallelism in Fortran and C/C++ programs
PIC - particle-in-cell, simulation method based on solving ion and electron trajectories
Vlasiator - a numerical model of near-Earth space based on the Vlasov theory of plasma

Variables

Additional subscripts e and i added to the following (where applicable) stand for electrons and ions respectively:
$\mathbf{B}, B$ - magnetic field and its magnitude
$\mathbf{E}, E$ - electric field and its magnitude
$\mathbf{V}, V$ - bulk velocity of plasma and its magnitude
$\mathbf{J}, J$ - current density and its magnitude
$\mathscr{P}, P$ - pressure tensor, scalar pressure
$\rho_m, \rho_n, \boldsymbol{\rho}_p, \rho_q, \rho_E$ - densities (ions + electrons for plasma by default) of mass, particle number, momentum, charge and total energy
$\mu_0 = 4\pi * 10^{-7}$ H m^{-1} (A^{-2} kg m s^{-2}) - permeability of vacuum
$\epsilon_0 = (\mu_0 c^2)^{-1} \approx 8.854 * 10^{-12}$ F m^{-1} (A^2 kg^{-1} m^{-3} s^4) - permittivity of vacuum

Chapter 1

Introduction

Prior to about 1950, the scientific method was primarily based on theory and experiments/observations. Around 1950 a new possibility emerged which could support the interpretation of observations [as used by e.g. Goodson et al., 1999] and the planning of experiments [such as Aad et al., 2010] and in some cases even replace either of them to a large extent. In this method - numerical modeling - equations governing the system are solved by a computer or, as it was called at the time, an automatic general-purpose programmable calculator. ENIAC (Electronic Numerical Integrator And Computer) was probably the first computer to be used for scientific computing and numerical modeling such as weather prediction, atomic-energy calculations, cosmic-ray studies, thermal ignition, random-number studies and wind-tunnel design [Weik, 1961]. ENIAC was able to calculate the ballistic trajectory of artillery ordnance over 2000 times faster than a skilled human using a desk calculator and could also be reprogrammed. Thus, instead of relying on physical experiments or significantly slower manual calculations, it was possible to find solutions in reasonable time, for example, to hydrodynamic problems in cases where an analytic solution was not known or does not exist.

This thesis describes the process of developing and applying numerical space plasma models for studying the near-Earth space and consists of an introduction and original research articles. Part of the work done in this thesis is applicable to a wide area of numerical modeling (especially Article I), hence examples of modeling in areas other than space weather are also discussed in the introductory part. In the rest of this chapter the topics of numerical modeling and space plasma are introduced. In Chapter 2 various analytic approximations to the fluid and electromagnetic (EM) equations of space plasma are discussed and the most often used numerical approaches for modeling space plasmas are presented. Based on the numerical model development experience obtained during this thesis Chapter 3 presents my preferred high level approach to solving a space weather modeling problem, starting from the search for existing solutions and codes to various optimization techniques. Chapter 5 discusses the testing, verification and validation of numerical space weather models that was done as part of this thesis. In Chapter 6 the space weather model GUMICS-4 is used to study large plasmoid formation in the Earth's magnetotail. Chapter 7 concludes the introductory part of the thesis.

1.1 Numerical modeling

The performance of computers has been growing exponentially probably since the first computers were constructed. During the last two decades this can be clearly seen from the performance of the 500 fastest supercomputers [TOP500.Org, 2013]. Largely due to this fact the importance of numerical modeling in science has also increased rapidly. Numerical modeling is especially important in space plasma[1] physics simply because there are no in-situ observations of space plasmas outside of the heliosphere and it is not feasible to study all aspects of space plasmas in a terrestrial laboratory. Even inside the heliosphere in-situ observations are scarce and the cost of obtaining them is usually measured in millions or more.

In this thesis numerical modeling is defined as solving a set of one or more equations which describe a physical system within a given volume. An analytic solution to the equations is seldom known or available, hence the equations are not solved symbolically as, for example, in Axiom [Daly, 2010] or Maple [Maplesoft, 2013] but instead a numerical approximation to the true solution is calculated.

Usually the system to be solved and the variables within the equations describing the system are continuous, such as the volume in space around the Earth where the magnetohydrodynamic (MHD) equations are solved. Discrete systems are also possible, such as Conway's Game of Life (GoL) [Gardner, 1970], different stock in the stock market or different species in population dynamics, but with the exception of GoL this thesis focuses on continuous systems.

As computers have a finite number of states, in order to be able to use them for solving a continuous system, the system must be discretized into a finite number of units that from hereinafter will be referred to as cells. Similarly in order to solve the equations describing a continuous system on a computer the equations must be discretized i.e. transformed into a representation in which a cell's state changes based on the states of one or more other cells of the system. This is true both for systems representing a volume of physical space as well as a volume of e.g. spatial frequency space used in spectral methods [Patera, 1984].

A program playing GoL is a simple example of a numerical model since the system is already discrete along with the laws governing its behavior, although a continuous version of GoL has also been invented [Rafler, 2011]. Despite its simplicity the GoL has been found useful also in e.g. statistical mechanics [Bak et al., 1989], quantum mechanics [Flitney and Abbott, 2005] and cryptography [Wang and Jin, 2012]. The original GoL is played in a 2d domain with rectangular cells, e.g. a chess board, and each cell has two possible states: alive or dead. At each turn of the game, or each time step of the simulation, the following set of rules decide whether the state of a cell in the simulation changes for the next time step:

1. If 3 out of the 8 nearest neighbors of the cell are currently alive the cell is alive in the next step

2. If instead 2 out of the 8 nearest neighbors of the cell are currently alive the cell's state stays the same in the next step

[1] In this thesis the term space plasma will be used to refer to plasmas both inside and outside of the heliosphere.

3. Otherwise the cell is dead in the next step

GoL exhibits many features present in more complicated numerical models and is useful for a general discussion on the subject without considering any specific problem in physics, chemistry, etc:

- The system is discrete, i.e. the modeled volume and the equations governing the system have been discretized and the simulation is advanced in discrete steps
- Each cell in the simulation stores the variable(s) representing the state of the modeled system, for example a quantity such as temperature, whose change in time is modeled
- At each step one or more functions collectively called a solver calculate how the variable(s) in each cell change for the next time step
- In order to calculate this change the solver requires data from the cell's neighbors within a finite distance from the cell itself

The procedure outlined above is quite specific to physical simulations and is distinct from other methods used for processing large amounts of data such as MapReduce [Dean and Ghemawat, 2004, Plimpton and Devine, 2011]. In MapReduce the user specifies two functions: 1) a map function which processes key/value pairs and generates intermediate key/value pairs, and 2) a reduce function which processes all intermediate values associated with the same intermediate key. For example, counting the number of nouns and adjectives in a large body of text is a MapReduce problem. For each chunk of input text the mapping step produces two intermediate key/value pairs which record the number of nouns and adjectives respectively found in the chunk of text. Subsequently two reduction steps produce the total number nouns and adjectives respectively for the whole body of text.

Table 1.1 shows an overview of the algorithms for a physical simulation and MapReduce when both systems are processed in parallel by multiple processes. The MapReduce program counts the number of nouns and adjectives in a large body of text while the simulation program plays the GoL. In the MapReduce program the mapping step for each chunk of data is independent of the other chunks, hence no exchange of information is required between processes doing the mapping. This also means that the time required for each map does not have to be constant as each process requests a new chunk whenever it has processed its previous chunk. The reduction steps are also independent of each other but usually the number of reduction steps is much lower than the number of mapping steps which can become a bottleneck for processing. In this case the total number of reductions is two, hence only two processes at most can process the reduce steps.

In GoL the grid cells are divided evenly between N processes and at each time step every process calculates the changes to its own cells for the next time step. In order for a process to be able to calculate the changes for its boundary cells, i.e. cells which have neighbors on other processes, each process must exchange the data of boundary cells between the other processes. Thus, in stark contrast to

Table 1.1: Overview of parallel data processing in a simulation such as GoL and in MapReduce such as counting the number of nouns and adjectives in a given set of data, see the text for details.

Time (arbitrary units)	Numerical simulation					MapReduce				
	Proc 1	P2	P3	P4	...	Proc 1	P2	P3	P4	...
1	solve					map	map	map	map	
2	exchange boundary data						map	map		
3	solve					map	map	map	map	
4	exchange boundary data					map	map		map	
5	solve					reduce	reduce			
6	exchange boundary data									
7	solve									
8	exchange boundary data									

MapReduce, the program speed is limited by the slowest process in the system. Also the communication speed between processes is essential for good performance and good parallel scalability.

1.2 Space plasma

> The world is made of 4 basic
> elements, earth, air, fire,
> water...
Today we call them "solid", "gas", "plasma", and "liquid" respectively.

Discussion on Slashdot

Matter that consists entirely or partially of free charged particles is called a plasma and is a fundamental state of matter along with gas, liquid and solid. The fraction of charged particles to neutrals in a plasma - the degree of ionization - varies highly but a good rule of thumb is that in a plasma the degree of ionization is large enough for collective EM behavior to be important [e.g. Koskinen, 2011] which can be as low as 0.01 % in the "neutral" hydrogen regions around galaxies [Peratt, 1996]. Almost all observable matter in the universe is in the plasma state, more specifically 99.999 % by volume [Peratt, 1996]. The range of parameters over which plasmas exists is huge compared to gases, liquids and solids. The density of a plasma can span 30 orders of magnitude, from 10^1 electrons per cubic meter in interstellar and intergalactic space to 10^{31} electrons per cubic meter in the center of the Sun. The magnetic field in a plasma varies almost as much from less than 10^{-10} T e.g. in the heliosheath [Burlaga et al., 2006], interstellar and intergalactic space to over 10^{11} T in magnetars [Duncan and Thompson, 1992]. The temperature of a plasma can also vary widely, from 10 K in interstellar space [Ferrière, 2001] to 10^8 K in some galaxy clusters [David et al., 1993] and centers of massive stars [The Imagine Team, 2013].

The characteristic time scales of heliospheric plasma range from electron oscillations of the order of 10^4 Hz [e.g. Koskinen, 2011] down to the 10^{-9} Hz (11 year) solar cycle [Schwabe, 1843]. Finally the spatial scales range from e.g. the Debye length of the order of 10^1 m in the Earth's magnetosphere to the size of the heliosphere (10^{13} m) and even larger astrophysical objects.

1.2.1 Near-Earth space plasma

The near Earth plasma environment is dominated by the solar wind which is a constant supersonic plasma flow, including a magnetic field [Coleman et al., 1960], outwards from the Sun. The Earth's magnetosphere is formed from the interaction of this wind flowing around the Earth and its intrinsic dipolar magnetic field. The basic configuration of the magnetic field in the magnetosphere in the polar (north-south) plane was first sketched by Dungey [1961] and is shown for example in Figures 1 and 2 of Article IV. In the sunward direction from Earth, i.e. day side, a shock is formed as the supersonic solar wind "hits" the Earth's dipole field and the dynamic pressure of the solar wind compresses the dayside magnetosphere to an approximate distance of 10 R_E from Earth, where R_E is the Earth's radius, approximately 6400 km. In the direction away from the Sun (night side) the magnetosphere extends for hundreds or even thousands of R_E [Dungey, 1965] forming the Earth's magnetotail. A more detailed picture of the magnetosphere is available from various sources [Christian, 2012, Michel et al., 2010] [Koskinen, 2011].

The kinetic energy flux of the solar wind at Earth varies mostly between 10^{-4} W/m^2 and 10^{-2} W/m^2 which is around six orders of magnitude less than the Sun's total irradiance at 1 au of about 1400 W/m^2. The solar wind plasma at Earth is highly collisionless as the mean free path of particles at 1 au is of the order of 1 au [e.g. Koskinen, 2011].

The plasma density in the magnetosphere ranges from as low as 10^{-2} H^+/cm^{-3} in the magnetotail lobes to over 10^2 H^+/cm^{-3} in the bow shock in front of the magnetosphere during space storms [Balch et al., 2004]. The kinetic energy of single particles in the magnetosphere varies from a few eV (around 30 km/s) in the dominant plasma population of the magnetotail lobes [Engwall et al., 2009] to over 10^6 eV of the highest energy particles in the Earth's radiation belts [Baker et al., 2013].

1.2.2 Space weather

Space weather refers to the day-to-day changes in plasma, magnetic field and other parameters in the near-Earth space, similarly to ordinary weather which refers to e.g. temperature changes in the atmosphere above ground or sea level. Space weather is driven by the energy of the solar wind which enters the Earth's magnetosphere mainly by magnetic reconnection through the interface between the solar wind and the magnetosphere called the magnetopause. The energy deposited into the magnetosphere is periodically released by magnetic reconnection in the Earth's magnetotail accelerating particles both towards and away from the Earth. The accelerated particles can follow the magnetic field lines all the way to the Earth's ionosphere leading to, for example, auroras.

Almost all space-based assets are located inside the Earth's magnetosphere and thus must endure the effects of space weather. For example weather, communication, global positioning and various other satellites are orbiting the Earth in regions of high energy ions and electrons such as the ring current and radiation belts. During strong magnetospheric storms particle energies in those regions can reach tens of MeV, i.e. several orders of magnitude higher than the energy of particles in a fusion reactor, posing a significant threat. Space weather can also cause adverse effects on the ground, for example, by disrupting radio-based communication and inducing large direct currents in pipelines and power transmission systems. One extreme example of such an event is the 1989 failure of the Hydro-Québec power system due to a severe magnetospheric storm which resulted in damages of several million $ [Bolduc, 2002] and a 9 hour blackout affecting millions of people. The increasing amount of infrastructure susceptible to space weather demand reliable forecasting of space weather and its effects both in space and on the ground.

Chapter 2

Introduction to modeling space plasma

Plasmas exist in the universe in a larger range of spatial, temporal, density, etc. scales than all other forms of matter which makes space plasmas an especially varied and challenging target for computational modeling. The spatial range includes everything starting from meter scales modeled for the electric sail [Janhunen and Sandroos, 2007] to hundreds of light years long intergalactic jets [Nishikawa et al., 1997] and galaxy formation on the scale of 10^5 light years [Wang and Abel, 2009]. Extreme astrophysical phenomena such as supernovae [Kotake et al., 2006], magnetars [Font et al., 2011] and black holes [Schnittman et al., 2006] are also being studied by numerical modeling. Heliospheric targets of modeling include solar flares, coronal mass ejections and the interaction of the solar wind with magnetospheres of strongly and weakly magnetized planets, non-magnetized bodies such as comets and man made objects.

2.1 Analytic approximations

This thesis deals with computational models that simulate a finite physical volume as a function of time, hence when writing equations it is assumed that all variables are functions of time (t) and space ($\mathbf{r}$), where applicable, and will not usually be written as such. In order to simplify the text it will be assumed that the system of interest has been discretized into cubic cells of equal size which cover the system. In practice both the cells' size and shape can vary, for example, when using adaptive mesh refinement (AMR), stretched grids and curvilinear coordinate systems. The term fluid will be used to refer to all flowing matter including gas, plasma or aerosols like volcanic ash in the atmosphere. The definition of a fluid can also include the charge state of the constituent particles so, for example, a plasma consisting of He^+ can be considered a different fluid than a plasma consisting of He^{2+}. Typically the equations discussed in the following sections are written separately for each particle species of interest, although formulations exist where a subset of the fluid equations are solved only for the whole fluid instead of different fluid species.

2.1.1 Continuity equation

The equations describing space plasma can be roughly divided into equations for the matter, which is assumed to be a fluid in this section, and equations for the electromagnetic fields. The simplest of the fluid equations, called the continuity equation, describes how the density of a fluid in each cell changes with time

$$\frac{\partial \rho_m}{\partial t} = -\nabla \cdot \boldsymbol{\rho}_p + Q, \tag{2.1}$$

where ρ_m is mass density and ρ_p is momentum density. The first term on the right hand side describes the flow of the fluid between cells based on the velocity $\mathbf{V} = \boldsymbol{\rho}_p/\rho_m$, i.e. advection. The second term (Q) is a source term that describes how much fluid is created or destroyed due to, for example, ionization, nuclear reactions or volcanic eruption and is equal to zero when mass in conserved. The velocity of the fluid can be constant, it can vary as a function of both time and space and it can also depend on the density of the fluid itself. The SILAM software [System for Integrated modeLling of Atmospheric coMposition, Sofiev et al., 2006] is an example of a computational model solving the continuity equation with a time-dependent velocity that is not affected by the fluid being advected. SILAM is used to model the flow of various particles in the atmosphere, for example, radioactive pollutants, pollen, dust, sea salt, etc. The velocity is taken from numerical weather prediction models such as the High Resolution Limited Area Model [e.g. Per Undén et al., 2002]. The inviscid Burgers' equation

$$\frac{\partial \rho(t, \mathbf{r}, \mathbf{V}(\rho))}{\partial t} = -\frac{\partial}{\partial x}\left(\frac{1}{2}\rho^2\right) \tag{2.2}$$

is an example of an advection equation in which velocity itself depends on density and leads to more complex phenomena such as shocks [LeVeque, 2002].

More than three dimensions

Density can also be a function of velocity $\rho(t, \mathbf{r}, \mathbf{v})$, in which case it is usually referred to as phase space density, and is a more complex approximation to the physical system than that given by Equation 2.1. Note the difference between $\mathbf{v}$ and $\mathbf{V}$: $\mathbf{v}$ is an additional dimension in the system, e.g. each simulation cell has a unique coordinate in phase (ordinary + velocity) space, while the bulk velocity $\mathbf{V}(\mathbf{r})$ is calculated from the velocities of all cells at a particular location of ordinary space $\mathbf{r}$. When the density is a function of velocity its behavior is determined by both the velocity and acceleration as opposed to only the velocity present in Equation 2.1. This is easier to visualize by marking the Nth time derivative of a variable by N dots above said variable. Excluding the source term Q, the behavior of $\rho(t, \mathbf{r})$ is defined by $\dot{\mathbf{r}}$ while the behavior of $\rho(t, \mathbf{r}, \dot{\mathbf{r}})$ is defined by $\ddot{\mathbf{r}}$ and $\dot{\mathbf{r}}$. Mathematically this can be continued up to arbitrarily high time derivatives but such formulations do not seem to have a physical foundation nor practical applications. On the other hand a 6d description of the phase space density of the fluid captures most effects present in fully ionized collisionless plasma, such as wave-particle interactions [e.g. Koskinen, 2011], that cannot be modeled with a 3d fluid description of plasma (Equations 2.1,

2.13 and 2.14). Such a kinetic formulation was first given for a neutral fluid in 1872 by L. Boltzmann:

$$\frac{\partial f(t,\mathbf{r},\mathbf{v})}{\partial t} = -\left(\mathbf{v}\cdot\frac{\partial f}{\partial \mathbf{r}} + \mathbf{a}\cdot\frac{\partial f}{\partial \mathbf{v}} + Q\right) \tag{2.3}$$

where Q is a source term describing the collisions between particles. A similar formulation for EM radiation, in which the source term describes emitted and absorbed photons, has been used to investigate radiative transfer [Harris, 1965] as well as to investigate neutrino transport in collapsing supernovae, albeit in less than 6d [Bruenn et al., 2013, Kotake et al., 2006]. In 1938 the Boltzmann equation was adapted to collisionless space plasma by A. Vlasov:

$$\frac{\partial f}{\partial t} = -\left(\mathbf{v}\cdot\frac{\partial f}{\partial \mathbf{r}} + \frac{\rho_q}{\rho_m}(\mathbf{E}+\mathbf{v}\times\mathbf{B})\cdot\frac{\partial f}{\partial \mathbf{v}}\right) \tag{2.4}$$

where the acceleration is given by the Lorentz force $F = q(\mathbf{E}+\mathbf{v}\times\mathbf{B})$ acting on the charged particles of the plasma. In principle both equations assume that the density represents a probability of finding particles within a volume of ordinary and velocity space. In computational plasma models based on Vlasov's equation (i.e. collisionless Boltzmann equation[Henon, 1982]) the density does not represent a probability but the actual amount of plasma within the (phase space) volume [e.g. Palmroth et al., 2013].

2.1.2 Remaining hydrodynamic equations

The continuity equation alone does not represent a fluid realistically unless the local mass fraction of the fluid of interest is much smaller than the total mass of all fluids, as it is, for example, in the case of dust and pollutants transported by the atmosphere. A more realistic behavior for a fluid is given by also considering Euler's momentum equation

$$\frac{\partial \boldsymbol{\rho}_p}{\partial t} = -\nabla\cdot(\mathbf{V}\otimes\boldsymbol{\rho}_p + IP), \tag{2.5}$$

where I is the unit dyad and $(\mathbf{V}\otimes\boldsymbol{\rho}_p)_i = \sum_j V_i\rho_{p_j}$, which describes how the fluid's momentum density changes in time due to, for example, the fluid's pressure. The continuity and momentum equations do not describe how the fluid's internal energy, or other thermodynamic quantities such as temperature, pressure and entropy, change in time but they are sufficient, for example, for deriving the shallow water equations [Randall, 2006] which are used for modeling rivers and coastal regions or similar systems such as some atmospheric phenomena. As with the continuity equation in which the velocity can be given externally, so can one or more of the thermodynamic quantities be given externally while solving the continuity and momentum equations. This could be the situation for example in an industrial process where gases flow and react in a system with externally set temperature.

The change in the internal energy of a fluid over time is given by the aptly named energy equation

$$\frac{\partial \rho_E}{\partial t} = -\nabla\cdot[\mathbf{V}(\rho_E + P)], \tag{2.6}$$

where ρ_E is total energy density and P is pressure. Along with an equation relating ρ_E, P and ρ_m called the equation of state, the energy equation completes the basic set of fluid equations. The Navier-Stokes formulation extend the above fluid equations by adding the effects of viscosity and heat conduction [e.g. Koskinen, 2011]. The fluid equations can be mathematically extended to more than 3d [used for example in Hong, 2013] but practical applications of such investigations seem extremely limited at the moment. Historically the continuity and momentum equations were derived by Euler from Newton's second law around 1750 [Christodoulou, 2007] but all of the fluid equations can also be derived from the Boltzmann equation (which itself can be derived from relativistic quantum field theory [Drewes et al., 2013]). When deriving the fluid equations from the Boltzmann equation one is confronted with an infinite series of equations, where the ith equation depends on the $(i+1)$th equation, which must be truncated at some point in order to obtain a tractable set of equations. A physically motivated truncation is to assume a collisional system, i.e. that the mean free path of particles between collisions is much smaller than the length scales that one is interested in. This allows one to calculate, for example, the pressure in the energy equation from mass and energy density using the ideal gas law.

2.1.3 Electromagnetic equations and their simplifications

The classical EM equations initially published by J.C. Maxwell in 1861, 1862

$$\nabla \cdot \mathbf{E} = \frac{\rho_q}{\epsilon_0} \tag{2.7}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{2.8}$$

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \mathbf{E} \tag{2.9}$$

$$\frac{\partial \mathbf{E}}{\partial t} = \frac{\nabla \times \mathbf{B}}{\epsilon_0 \mu_0} - \frac{\mathbf{J}}{\epsilon_0} \tag{2.10}$$

are rarely used in their complete form in computational plasma models as such a model would in many cases be computationally prohibitively expensive and one is also seldom interested in all possible plasma and EM phenomena simultaneously. For example, when modeling the Earth's magnetosphere, EM radiation can be ignored by removing the displacement current term $\partial \mathbf{E}/\partial t$ from Equation 2.10 transforming it into Ampère's original circuital law $\mathbf{J} = \mu_0^{-1} \nabla \times \mathbf{B}$. On the other hand, the displacement current term is required if the speed of Alfvén waves [Alfvén, 1942] is not to exceed the speed of light [J.P. Boris, 1970]. In this approach the speed of light can also be reduced which can decrease the time to solution to less than 1/10 in an MHD simulation of the Earth's magnetosphere [Gombosi et al., 2002]. In the above cases the electric field is calculated from other system variables by using a generalized Ohm's law which is discussed in Section 2.1.5. This formulation corresponds to the assumption that the average charge densities of ions and electrons are equal $\rho_{q_i} = \rho_{q_e}$ and hence on average the plasma is not charged i.e. it is quasi-neutral [Ledvina et al., 2008]. This assumption is valid on spatial scales larger than the Debye length which is of the order of 10 m in the Earth's magnetosphere. EM radiation can also be ignored by splitting $\mathbf{E}$ into longitudinal E_{lon} and transverse

$\mathbf{E}_{trans}$ components with respect to $\mathbf{B}$ and neglecting $\partial\mathbf{E}_{trans}/\partial t$ in Equation 2.10, which also removes relativistic phenomena [Ledvina et al., 2008].

When the effects of EM radiation cannot be ignored various simplifying assumptions are still possible. If radiation created by a medium affects the medium itself significantly, as is the case in some shocks, one can use a diffusion approximation for radiation which adds radiation energy terms to the fluid's momentum and energy equations and assumes that energy does not escape the shock region [Sen and Guess, 1957]. An even more complete description of the propagation of radiation in a medium is given by the radiative transfer approximation [Chandrasekhar, 1960] which is used, for example, when studying stellar atmospheres and other astrophysical problems [Abel et al., 1999, Schnittman et al., 2006]. When studying the EM scattering problem, i.e. the scattering of EM radiation by particles such as aerosols, Maxwell's equations are solved without sources, i.e. with $\rho_q = 0$ and $\mathbf{J} = 0$ [Kahnert, 2003].

In a fully kinetic simulation in which the evolution of both ions and electrons is solved the quasi neutrality assumption usually does not hold and hence the electric field created by non-zero total charge density should be taken into account. When the effects of the magnetic field or its change as a function of time are not significant such as in the case of the electric solar wind sail the electrostatic approximation of Maxwell's equations can be used [Janhunen and Sandroos, 2007]. In this approximation neither E nor B are modeled directly but instead B is set externally, usually to zero, and E is calculated from the charge density as $\mathbf{E} = \nabla\phi$, where ϕ is calculated from $\nabla \cdot (\nabla\phi) = \rho_q/\epsilon_0$. On the other hand when the evolution of E and B are significant but the total charge in the simulation is constant one does not need to solve the above equation for the electric field but only the time evolution of E and B from Eqs. 2.12 and 2.10 [Pohjola and Kallio, 2010]. When none of the above assumptions can be made the full set of Maxwell's equations are used [as e.g. in Daughton et al., 2006].

There are also esoteric formulations of Maxwell's equations in which magnetic monopoles are allowed to exist (i.e. $\nabla \cdot \mathbf{B} \neq 0$), for example, in MHD where this formulation is used for advecting the non-physical divergence of B created by the numerical solution out of the simulation [Powell, 1997] and for keeping the numerical solution positive and conservative [Janhunen, 2000].

2.1.4 Plasma equations: coupled matter and EM equations

Electric and magnetic fields do not affect neutral fluids, such as the Earth's atmosphere below some 50 km, although large EM fields created by e.g. powerful lasers can ionize a neutral gas into the plasma state. Even in the case of plasma the effect of the magnetic field can be ignored if the plasma's Lundquist parameter L_u a.k.a. Lundquist number $S(\equiv L_u)$ is much less than unity

$$S = \sqrt{\frac{\mu_0}{\rho_m}} L\sigma B \ll 1 \tag{2.11}$$

where L is a characteristic length of the plasma and σ is its conductivity. In the Earth's atmosphere S falls below unity at a height of about 100 km and less from

the sea level. The Earth's and other planets' atmospheres and hydrospheres are the only known domains in the universe in which the effect of the magnetic field can be ignored [Peratt, 1996].

In probably all computational space plasma models which solve the evolution of the magnetic field self-consistently the evolution is described by Faraday's law of induction

$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \mathbf{E}. \tag{2.12}$$

In models which do not solve the full set of EM equations, for example if it is assumed that $\partial \mathbf{E}/\partial t = 0$, the electric field is approximated by a generalized Ohm's law which is discussed in Section 2.1.5. From hereinafter this approximate electric field will be denoted by $\mathbf{E}_F$. A self-consistently evolving magnetic field is not always required. In that case $\mathbf{B}$ can be approximated by a constant value [e.g. in Janhunen and Sandroos, 2007], by a statistical model [e.g. in Sheldon et al., 1998] or by a self-consistently calculated field from another model as was done in Article II.

Simplifying assumptions

In the MHD, hybrid Vlasov and hybrid PIC modeling approaches discussed in the following sections the mass of electrons is assumed to be zero which is a reasonable approximation in many cases as the mass of ions is at least three orders of magnitude larger than the electrons. Together with the quasi neutrality assumption, the assumption of massless electrons removes all high frequency and small spatial scale plasma physics from the modeled system. These assumptions also create a lower limit to the size of cells that can be used in a model in order to obtain a reliable solution. Additionally the formation of unphysical EM fields is also possible in cases where ∇P_e is significant but is not modeled due to the above assumptions [Ledvina et al., 2008].

Magnetohydrodynamics

When the effect of the magnetic field must be taken into account the momentum and energy equations of hydrodynamics can be extended with magnetic field terms:

$$\frac{\partial \boldsymbol{\rho}_p}{\partial t} = -\nabla \cdot \left[\mathbf{V} \otimes (\boldsymbol{\rho}_p) + I \left(P + \frac{B^2}{2\mu_0} \right) - \frac{\mathbf{B} \otimes \mathbf{B}}{\mu_0} \right] \tag{2.13}$$

$$\frac{\partial \rho_E}{\partial t} = -\nabla \cdot \left[\mathbf{V} \left(\rho_E + P + \frac{B^2}{2\mu_0} \right) - \frac{\mathbf{B}(\mathbf{V} \cdot \mathbf{B})}{\mu_0} \right]. \tag{2.14}$$

MHD assumes that the fluid is close to thermodynamic equilibrium and that the fluid pressure is isotropic, which are valid assumptions when the fluid is collisional [Ledvina et al., 2008]. The simplest approximation for the electric field in MHD is the ideal MHD approximation

$$\mathbf{E}_F = -\mathbf{V} \times \mathbf{B}. \tag{2.15}$$

Vlasov

The Vlasov equation for plasma is simpler than the system of equations for MHD in the sense that only one equation for each fluid species, instead of a coupled system of three, describes the motion of the fluid. In the simplest case called hybrid Vlasov only ions are modeled using the Vlasov equation and the evolution of the magnetic field is given by the induction equation almost exactly as in ideal MHD, i.e. $\partial \mathbf{B}/\partial t = -\nabla \times \mathbf{E}_F$ where $\mathbf{E}_F = -\mathbf{V}_i \times \mathbf{B}$ and $\mathbf{V}_i$ is the bulk ion velocity. It is important to note that the electric field $\mathbf{E}_F$ in the induction equation need not be identical to the electric field $\mathbf{E}_L$ in the Vlasov equation that accelerates the fluid through the Lorentz force

$$\mathbf{F} = q_i(\mathbf{E}_L + \mathbf{v}_i \times \mathbf{B}) \tag{2.16}$$

where $\mathbf{v}_i$ is the velocity of the fluid in the 6d (phase space) cell i of the simulation and q_i is the charge of the fluid in that cell. $\mathbf{E}_L$ must include at least the $\mathbf{J} \times \mathbf{B}$ term from the generalized Ohm's law (see Section 2.1.5), where $\mathbf{J} = \mu_0^{-1} \nabla \times \mathbf{B}$, as otherwise no bulk force will affect the plasma [Karimabadi et al., 2004]. To put it another way, without the $\mathbf{J} \times \mathbf{B}$ term the plasma will always be stationary on average from the point of view of the Lorentz force, i.e. $\mathbf{v}_i$ will be relative to the average velocity of plasma instead of the laboratory frame:

$$F = q_i(\mathbf{E}_L + \mathbf{v}_i \times \mathbf{B}) = q_i(-\mathbf{V}_i \times \mathbf{B} + \mathbf{v}_i \times \mathbf{B}) = q_i((\mathbf{v}_i - \mathbf{V}_i) \times \mathbf{B}) \tag{2.17}$$

which, for example, makes the solar wind pass right through the Earth's dipole field because in such case $|\mathbf{v}_i - \mathbf{V}_i| \ll |\mathbf{v}_i|$ and the force exerted on ions is almost insignificant in comparison. In a full Vlasov simulation modeling both ions and electrons the EM equations are solved, for example, in one of the ways described in Section 2.1.3.

Discrete particles

In the particle-in-cell method plasmas are modeled as discrete particles which represent ions or electrons. Since tracking every physical particle would be computationally prohibitively expensive each simulation particle represents a number of real particles of the same species with identical position and velocity, i.e. the mass of simulation particles is much larger than in reality while their charge to mass ratio is correct or at least realistic. For example in the hyb model [Kallio and Janhunen, 2003] each particle typically represents an ion number density of the order of 10^5 m^{-3} which roughly translates to a simulated particle mass of 10^{20} u (unified atomic mass unit) or 0.1 mg assuming cells of 100^3 km^3 in size. In hybrid PIC the magnetic field is solved using Equation 2.12 and the particles are propagated using Equation 2.16. The local spatial averages (bulk values from hereinafter) of several plasma parameters are required in Equation 2.12. For example, if Equation 2.15 is used to represent the electric field in Equation 2.12, the bulk plasma velocity in each cell must be computed. The bulk quantities of plasma are interpolated onto cells from nearby particles, in the simplest case, by assigning each particles' velocity to the cell inside of which the particle is currently located. A higher-order method of interpolation is given e.g. in Kallio and Janhunen [2003]. Unless the number of

particles in each simulation cell is large enough the simulation can become unstable or produce unphysical results. On the other hand too many particles only make the simulation unnecessarily computationally expensive, hence the number of particles is usually kept at a constant value in each cell. The number of particles can be adjusted at runtime by splitting and joining them, for example as described in Kallio and Janhunen [2003] where the total mass, momentum and kinetic energy of the particles is preserved by the procedure. The method also does not move the center of mass of the particles and preserves the total angular momentum of the particles with respect to the center of mass before and after splitting.

Effects of modern physics

The fluid and EM equations presented in this section do not include quantum effects nor the effects of special or general relativity. Typically relativistic effects are important in astrophysics, while quantum effects potentially provide a fundamental explanation for magnetic reconnection [Treumann et al., 2012]. The following works, for example, and references therein present relativistic treatments of the fluid and EM equations along with various applications: MHD Baumgarte and Shapiro [2003], Vlasov Andréasson [2011] and PIC Fonseca et al. [2002].

2.1.5 Generalized Ohm's law

The specific form of the electric field in the induction equation $\partial \mathbf{B}/\partial t = -\nabla \times \mathbf{E}_F$ varies depending on the assumptions made about the system. A generalized Ohm's law which contains the most important terms for a space plasma is [e.g. Koskinen, 2011]

$$\mathbf{E}_F = -\mathbf{V}_i \times \mathbf{B} + \frac{\mathbf{J}}{\sigma} + (ne)^{-1}\left(\mathbf{J} \times \mathbf{B} - \nabla \cdot \mathscr{P}_e + \frac{m_e}{e}\frac{\partial \mathbf{J}}{\partial t}\right) \tag{2.18}$$

where each term has the following effect [Drake, 1995, Ledvina et al., 2008]:

- Assuming zero electron mass or neglecting $\partial \mathbf{J}/\partial t$ removes electron inertia effects from the system that correspond to length scale of electron skin depth and time scale of electron plasma frequency.
- Neglecting $\mathbf{J} \times \mathbf{B}$ removes from the system phenomena on spatial scale of ion skin depth and temporal scale of ion plasma frequency.
- Assuming isotropic electron pressure $\nabla \cdot \mathscr{P}_e = \nabla P_e$ removes effects of non-Maxwellian electron populations and neglecting electron pressure completely removes effects on the spatial scale of effective ion gyro radius from the system.
- When all of the above assumptions hold the generalized Ohm's law reduces to resistive MHD: $\mathbf{E}_F = -\mathbf{V} \times \mathbf{B} + \mathbf{J}\sigma^{-1}$, where σ represents anomalous resistivity in a collisionless system due to e.g. current driven or two-stream instabilities [Drake, 1995, Yamada et al., 2010].
- Finally, very large magnetic Reynolds number, i.e. very large conductivity, in resistive MHD leads to ideal MHD: $\mathbf{E}_F = -\mathbf{V} \times \mathbf{B}$

2.2 Numerical approaches

There are many numerical approaches for modeling of space plasma. They range from various ways of discretizing the simulated volume and the mathematical equations to different representations of the simulated quantities and the types of solvers used for computing the solution. Here some of the most often used approaches are described.

2.2.1 Discretizing the volume

When solving equations in a physical volume in general, two methods exist for discretizing the volume. In the one employed e.g. in numerical space weather prediction the volume is discretized as in GoL, that is, the location of each cell in the grid is fixed. This method is called Eulerian as the variables on the grid are defined as functions of position [Batchelor, 2000]. In a Lagrangian method the variables are defined as functions of parcels/particles/cells which can move freely within the simulated volume. For example in smoothed particle hydrodynamics (SPH) the fluid is represented by a collection of fluid particles and a fundamental problem then is how to calculate e.g. the density of the fluid in a specific location [Price, 2012]. In principle a Lagrangian method does not require the use of a grid (in which case the method is grid-free) but in many cases this is necessary for obtaining acceptable computational performance as e.g. in SPH the behavior of a fluid particle depends on its nearest neighbors and whose data must be found in memory. The speed of this calculation is often optimized by interpreting the particles and their nearest neighbors as an unstructured moving grid. In PIC formulation the plasma is represented by a collection of particles while the electric and magnetic fields are calculated on a grid based on fluid quantities calculated from the particles. In a semi-Lagrangian scheme the particles are periodically interpolated into a fixed grid after which the particles are recreated at points defined by the grid and their motion simulated further.

2.2.2 Discretizing the equations

Three widely used approaches for modeling fluid equations numerically are the finite difference, finite element and finite volume methods [FDM, FEM and FVM respectively, see e.g. Eymard et al., 2000, LeVeque, 2002].

In an FVM simulation cells represent small control volumes and changes in variables of the system are calculated via fluxes through the faces between cells. This leads to perhaps the most attractive feature of FVM which is the conservation of the modeled variables by definition as for any flux calculated between two cells the exact same quantity is removed from one cell and added to the other. Flux conservative equations of the form $\partial\rho/\partial t = -\nabla \cdot F$, where F is the flux, express this mathematically. FVM also lends itself naturally to conservative algorithms which correctly capture the physics of shocks such as their speed of propagation and jumps in density, energy, etc. across the shock. FVM is also easy to extend to non-uniform grids, for example when using AMR as was the case in Article I. The GUMICS

and Vlasiator models, which are the topic of this thesis in Articles II and III, are implemented using FVM.

The FEM method is based on a variational formulation in which the original equation is solved in its weak form, i.e. integrated with nearly arbitrary test function(s) [e.g. Eymard et al., 2000]. For example [Strang and Fix, 1988] the weak form of

$$-\frac{d}{dx}\left(c(x)\frac{du}{dx}\right) = f(x), \tag{2.19}$$

using test function(s) $v(x)$ is

$$\int_0^1 c(x)\frac{du}{dx}\frac{dv}{dx}dx = \int_0^1 f(x)v(x)dx. \tag{2.20}$$

In Galerkin's method the discrete result is approximated by a linear combination of piecewise polynomial trial functions $\phi(x)$:

$$U(x) = \sum_{i=1}^{N} U_i\phi_i(x) \tag{2.21}$$

where the values U_i are obtained from equations representing the discretized test function(s) $V_i(x)$:

$$\int_0^1 c(x)\left(\sum_{j=1}^{N} U_j\frac{d\phi_j}{dx}\right)\frac{dV_i}{dx}dx = \int_0^1 f(x)V_i(x)dx. \tag{2.22}$$

Often the same polynomial is used for both the trial and test functions. When using high order polynomials FEM can be much more accurate than FDM or FVM but is also more difficult to implement. An example of using an FEM method to solve the Poisson's equation is presented in Section 3.4.1.

For completeness the FDM is also mentioned. In FDM the derivatives of the governing equations in each cell are replaced by finite differences through the equations' Taylor expansions in space around each simulated point [Eymard et al., 2000]. Often this approach gives methods that look very similar to related finite volume methods, however FVM is more robust when discontinuities are present [LeVeque, 2002]. Conservative formulations of the finite difference method also exist [e.g. Morinishi et al., 1998].

2.2.3 Cell, face, edge centered variables

In FVM cell variables represent volume averages of the quantities being modeled. In some cases it is important to make a quantity divergence free as dictated by the analytic approach, for example $\nabla\cdot\mathbf{B} = 0$ in MHD or $\nabla\cdot\mathbf{V} = 0$ in incompressible flow. This can be accomplished, perhaps most easily, by storing the relevant vector fields B and V on cell faces instead of cell centers. Such formulation was first used for HD by Harlow and Welch [1965] and for Maxwell's equations by Yee [1966] and is also used e.g. in Vlasiator [von Alfthan et al., 2013b]. In general these, along with many other, formulations seem to be a specific application of discrete exterior calculus

which will not be discussed further in this work. See Hirani [2003] for a theoretical treatment on the subject with many references to more practical applications such as Mattiussi [1997].

2.2.4 Explicit and implicit solvers

Numerical methods for solving ordinary or partial differential equations can be roughly divided into two classes. In the first one the solution at the next time step depends only on the current or previous steps and in the second one the solution also depends on the future solution itself. Press et al. [2007] present an example where the explicit and implicit Euler schemes are applied to the equation $dy(t)/dt = -cy(t)$ with $c > 0$ resulting in

$$y(t + \triangle t) = y(t) - y(t)c\triangle t \tag{2.23}$$

and

$$y(t + \triangle t) = y(t) - y(t + \triangle t)c\triangle t \tag{2.24}$$

respectively. The latter implicit formulation is stable for all step sizes $\triangle t$ and, despite being computationally more demanding for a single time step and harder to implement, can lead to significantly shorter times to solution than the explicit formulation. The accuracy of an implicit solution, computed using longer time steps than would be allowed by an explicit solver, is usually worse than the explicit solution computed with shorter time steps. Tóth et al. [2006] speed up a parallel space weather model by applying a combination of explicit and implicit MHD solvers. While implicit solvers seem well suited also for modeling the Earth's magnetosphere none of the numerical space plasma models developed at FMI use implicit solvers at the moment and hence they will not be discussed further in this work.

2.2.5 Multiple combined / coupled models

Multiple physical approximations can be combined into one numerical model which from hereinafter will be referred to as a multi physics model. Such a model is difficult to investigate analytically as the usual methods of analysis, for example calculating the dispersion equation for a plane wave propagating in the plasma [e.g. Koskinen, 2011], assumes a homogeneous description of the system using either MHD or Vlasov theory, for example. On the other hand a multi physical description would have e.g. the MHD equations in effect in one part of the system and the Vlasov-Maxwell equations in effect in another part, and possibly even some type of a combination of both along the interface between the two regions. I am not aware of analytical treatments of such a system.

The largest purely practical benefit of a multi physics model is the possibility of solving large parts of the system using a physically less accurate and computationally much less demanding description of the plasma, thereby reducing the computational cost of the model significantly. Another advantage is not having to rely on artificial or non-interacting boundary conditions in the physically more accurate but small region but instead being able to simulate a much larger system and have e.g. interacting boundary conditions in the physically more accurate region.

The Space Weather Modeling Framework [Tóth et al., 2012] is an example of a multi physics model for space plasmas. In addition to modeling the Earth's magnetosphere using MHD it can model the inner magnetosphere using a kinetic radiation belt module and several kinetic inner magnetospheric modules that solve the Boltzmann equation in two or 3d. Space plasma models combining the MHD and hybrid PIC descriptions have also been developed [e.g. Sugiyama and Kusano, 2007]. In the above models the region(s) where a physically more accurate description of the plasma is used cannot be changed while the simulation is running i.e. runtime adaptive physics are not supported. While numerical models supporting runtime adaptive physics have been used for modeling neutral fluids for almost a decade [Schwartzentruber and Boyd, 2006, Tiwari et al., 2009, Wijesinghe and Hadjiconstantinou, 2004], and have also recently been applied to laboratory plasmas [Kolobov and Arslanbekov, 2012], a space weather model supporting such a feature does not exist yet to my knowledge.

2.2.6 Global versus local numerical models

The term global model is usually used to describe numerical models which solve a problem in the entire volume of the system of interest including the true number of dimensions in the system. For example a global magnetospheric MHD model solves the system in 3 (ordinary space) dimensions and includes a large enough volume around the Earth to account for magnetotail dynamics such as substorms which also affect the Earth's ionosphere. On the other hand within a study of the magnetosphere the ionosphere can be approximated well (in spatial scales that are relevant to global magnetospheric phenomena) with only a 2d model which is an approximation used in most global magnetospheric MHD models (Article III). As this thesis deals mostly with global MHD models of the Earth's magnetosphere, from hereinafter the term global (MHD) model will be used when referring to global magnetospheric MHD models. In practice the sunward boundary of a global model can be just outside of the bow shock. Usually the boundary is located a bit over 30 R_E from the Earth. On the other hand the dynamics of the magnetotail, for example plasmoids (Articles III and IV), can extend to over 200 R_E downstream from the Earth while still affecting the ionosphere and hence should be included in a global model. From the point of view of magnetospheric physics Daughton et al. [2006] is an example of a local simulation: Only 2 dimensions are modeled; the size of the simulated system, when scaled to magnetospheric parameters, is of the order of 1 R_E; the intrinsic dipole field of Earth is not included; and the coupling of the magnetosphere and ionosphere is not modeled. Blanco-Cano et al. [2009] is an example of a semi-global simulation in which only 2 dimensions are modeled, the magnitude of the Earth's dipole is scaled by approximately a factor of 0.1, and the inner boundary of the magnetosphere is a perfectly conducting sphere.

Chapter 3

Developing a numerical model

> Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it.
>
> Brian W. Kernighan and P. J. Plauger in The Elements of Programming Style

This chapter introduces the software development side of space plasma modeling starting with an overview of my preferred solution to handling a particular modeling need. Testing, verifying and validating a numerical model, which is essential for almost any work carried out using that model, is discussed in Chapter 5.

Overview

Given a particular scientific problem which has already been determined to require a numerical model, the approach to solving that problem can be divided into several distinct steps. As this thesis is about developing high performance computational numerical models the following list, based on personal experience, states possible steps towards increased speed, i.e. decreased time to solution. One should move from one item to the next only if the program is clearly not fast enough for given modeling requirements.

1. Search for existing software in public software repositories

2. Implement a well scaling algorithm serially using an easy language, effort is usually within the range 10-1000 work days

3. Parallelize for distributed memory machines, speedup factor of 10-1000, effort a factor of 0.01-100 with respect to initial development

4. Optimize code for speed, speedup factor of 2-20, effort a factor of 2-100 with respect to initial development

5. Re implement using a faster language, speedup factor of 2-10, effort a factor of 0.1-10 with respect to initial development

3.1 Repository search

Regardless of the nature of the numerical modeling problem chances are high that a program solving it or a closely related problem has already been developed. For example there exist a multitude of freely [1] available programs for solving heliospheric and astrophysical problems using HD, MHD and PIC such as AstroBEAR [Cunningham et al., 2009], Athena [Stone et al., 2008], NDSPMHD [Price, 2012], RAMSES [Teyssier, 2002], SAMRAI [Wissink et al., 2001] and PLUTO [Mignone et al., 2007] just to name a few. The same is true for grid libraries providing structured or unstructured grids with support for one or more of: AMR, load balancing, remote neighbor updates, etc. Examples include DCCRG (Article I), p4est [Burstedde et al., 2011], deal.ii [Bangerth et al., 2007], libMesh [Kirk et al., 2006], PARAMESH [MacNeice et al., 2000] and Chombo [e.g. Van Straalen et al., 2011]. Consequently before starting to develop any non-trivial software one should spend some time searching the literature and software repositories for suitable programs or libraries. Furthermore non-trivial software likely consists of several independent parts and for each part a library might already have been developed.

Assuming the third party library can be trusted, i.e. it has been properly tested and verified (Chapter 5), and taken advantage of relative easily, effort can be instead spent on studying the physical problem. Reusing others' code is also advantageous because it lowers the error rate in programming and the overall software development effort, although this depends highly on whether components can be reused without modification [Thomas et al., 1995]. Typical places to check when searching for software include program repositories such as Github, Bitbucket, Gitorious, SourceForge, more specialized collections such as the Computer Physics Communications program library and, of course, Wikipedia (e.g. list of FEM software).

3.2 Simple working (serial) program

> Premature optimization is the root of all evil (or at least most of it) in programming
>
> Donald E. Knuth in his 1974 Turing award lecture

The techniques, methodologies and best practices of regular software development are out of scope of this thesis which concentrates on development and usage

[1] Free as in speech not beer (https://www.gnu.org/philosophy/free-sw.html)

of physical computational models, but some basics are briefly mentioned in Section 3.2.2. It is assumed that the program being developed is not trivial i.e. that the programmer has not written a similar program previously and that during its lifetime the program will be used for a much longer time than what was spent developing it. Optimizing a program which works incorrectly is of very little use thus the first priority after determining that a suitable program does not already exist should be to implement a working program in as simple way as possible, albeit with some constraints detailed in Section 3.2.1. The simple version of a program is assumed to be serial, which might not be true if the programming language supports distributed memory parallelism natively as do, for example, Erlang [Scalas et al., 2008], Coarray Fortran and Unified Parallel C [UPC Consortium, 2005]. In this case distributed memory parallelism can be taken advantage of automatically which can simplify the development procedure significantly. The implementation language might also be constrained by third party dependencies, for example the language might be dictated by an existing library that has been identified as very useful for solving the problem at hand, or a sufficiently large part of it.

3.2.1 Use a proper algorithm

During the entire lifetime of scientific software it is quite improbable that one will never need a significantly shorter time to solution, hence future speedup should be considered already at this stage of development. This is especially true when selecting the underlying algorithm(s) that will be used as some are inherently more scalable both to larger systems and from the point of view of distributed memory parallelism. For example a method for calculating the N-body interaction between bodies/particles in $O(N \log N)$ time, where N is the number of particles, has been known for 20 years [Warren and Salmon, 1993] hence there is very little point in implementing a $O(N^2)$ method, even in a serial program.

Simulation speed can also be increased by concentrating resolution to specific areas of the simulation by using a stretched [Raeder et al., 2008] or a curvilinear grid [Lyon et al., 2004] or by using AMR [Janhunen et al., 2012, Powell et al., 1999, Wise et al., 2012], which in some cases can yield a larger speedup than any other method. In the example presented in Wise et al. [2012], where star formation in a galaxy was modeled, using the maximum resolution everywhere would have increased the total number of cells by a factor of 10^{10}. Another way of speeding up a simulation is to use temporal subcycling, i.e. a non-uniform time step, in the simulation [Janhunen et al., 1996, 2012]. These methods are especially suited to problems in which the system contains localized gradients in one or more simulated variables or systems where one is most interested in the solution only in a small fraction of the total simulated volume, such as spacecrafts' trajectories.

Other algorithmic improvements include (see Section 2.2 for a discussion on the following): 1) Implicit or semi-implicit solvers which are not limited by the CFL condition [Courant et al., 1967] thus allowing the simulation to take significantly longer time steps with a moderate increase in computational demands; 2) Using a computationally much less demanding solver in areas where simpler physics are applicable or in which a physically more correct result is not as important. When algorith-

mic optimizations cannot anymore be expected to increase simulation speed by a factor of 10 or even 100 the next step should be distributed memory parallelization discussed in Section 3.4.

3.2.2 Software development

> All problems in computer science can be solved by another level of indirection
>
> David Wheeler, inventor of the subroutine

Everyone who has developed programs for scientific work can imagine what would be required from someone else's software in order to rely on it in one's own work, i.e. what is required of quality software. The basic attributes of quality software have been summarized e.g. by Adrion et al. [1982] but some of them, especially understandability and maintainability, have been underrepresented in scientific software. For example while the quality of numerical climate models from the point of view of the developers is higher than that of free and open source software in general, it is possible that this is due to the fact that scientists do not consider basic attributes of software quality such as usability and transportability relevant [Pipitone and Easterbrook, 2012]. On the other hand the static and dynamic analysis of millions of lines of code in Hatton [1997] paints a much less flattering picture of scientific software. One must also keep in mind that scientific software development is not directly comparable to industrial and commercial software development as scientific software is written to explore the unknown and it is usually impossible to provide complete software requirements in advance [Pipitone and Easterbrook, 2012].

Software development in general is a core topic of computer science and will not be covered here. For example at the time of writing the following topics, in no particular order, were taught in the courses Elements of Software Construction, Software Engineering Concepts at MIT OpenCourseWare: specifications and invariants, testing, test-case generation and coverage, abstract data types and representation independence, design principles, software process and lifecycle, requirements and specification, formal analysis and reviews, quality management and assessment, commercial off-the-shelf and reuse, evolution and maintenance. What still does seem to be missing from computer science classes is the development of free and open source software (FOSS) over the Internet by a loose collection of volunteers and professionals. The lack of experience in FOSS development might explain in part the large number of quite similar astrophysics software that has been developed (Section 3.1). Even though the features of those programs do not overlap exactly, in each case the next model might have been developed only because of the lack of usability, understandability or maintainability of the previous models or the not-invented-here philosophy. Raymond [1999] is one of the earliest analyses of the bazaar-style approach to software development, in which software is developed over the Internet in view of the public.

A more specific example is provided by the events leading to the development of dccrg (Article I). Before starting to develop dccrg I did do a repository search of existing software with similar features but no such library seemed to exist. The p4est library, for example, did not seem to support transparent remote neighbor updates or arbitrary data in grid cells and definitely did not seem as easy to use as e.g. Figure 4 of Article I. Transparency in this case means that the user has to write as little additional code as possible in order to transfer cell data between processes. Ideally no additional code would be required in the main loop of the program but at the moment one line is required when using dccrg, for example, in Figure 4 or Article I. Extending p4est to include the desired features seemed to require more effort than developing a library with the necessary functionality from scratch. Some features of p4est, or lack of them, were probably due to a different target audience as p4est seems to be geared more towards FEM simulations. Also some features available in dccrg would be more difficult to implement in p4est due to the C programming language lacking built-in support for object oriented and template programming.

3.3 (Super) Computers as a hierarchy

Modern computing hardware is highly hierarchical from the point of view of speeding up an application. While most of the hierarchy is not readily visible to the programmer, i.e. any application will run on a single computer, the performance of an application can be very bad compared to an application that takes the hierarchical nature of hardware into account either explicitly by using e.g. vectorization and threading (Sections 3.6 and 3.7) and/or implicitly by using cache-oblivious algorithms (Section 3.5).

Starting from the level of the hierarchy with the largest performance, first are the registers of each CPU core which store the machine instructions and data to be processed. The size of registers (of the order of 10 kB in total per CPU) is orders of magnitude lower compared to the amount of data any non-trivial program processes. The latency of registers, i.e. the time it takes to operate on data stored in them, is the smallest as the data are, by definition, "instantly" available for processing and are processed with a speed of the order of 1 operation per CPU clock cycle (see Section 3.6 for a better estimate). Next in the hierarchy are three levels of CPU cache called L1, L2 and L3 with total sizes of roughly 100 kB, 1 MB and 10 MB per CPU respectively and latencies of about 5, 10 and 40 clock cycles or 2.5, 5 and 20 ns assuming a 2 GHz clock frequency [Levinthal, 2009]. Typically each CPU core has its own L1 cache while the L2 cache is shared between few cores and the L3 cache is shared between all cores located on the same die. Next in the hierarchy is the main memory with a size of roughly 16 GB per CPU and a latency of about 50 to 100 ns.

Shared memory parallelization can add additional latencies as data modified in the L1 cache of one core must be transferred to the L1 caches of other cores before being available to them. The same is true for L3 caches of different CPUs where it might take even several hundred cycles or 100 ns for a remotely modified value to become usable in a local L3 cache. In a Non Uniform Memory Access (NUMA) systems the same holds true also for accesses to main memory from different cores adding one more level to the hardware hierarchy.

Main memory is the last level of the hierarchy that is transparent to the programmer as the next level - distributed memory parallelism - practically requires that the necessary data be sent explicitly by the programmer from the main memory of one machine or supercomputer node to another (see Section 3.4). Nevertheless the idea is still the same, i.e. data that is not available in local memory, or that has been updated in another node, must be transferred to local memory via a physical network before being used. At this level the latency has already increased to microsecond range, for example, being about 10 μs in Cray's Seastar network [Brightwell et al., 2005]. The Seastar network represents a 2d or 3d grid with periodic boundaries, i.e. each node connected to four or six others. This adds another level in the hardware hierarchy as data that resides on a node not directly connected to the current node must be transferred through several Seastar links which increases latency compared to transferring data from a neighboring node over a single link. Optimizing data access at this level can increase the effective network bandwidth of a program by over 50 % [von Alfthan et al., 2013a]. Cray's latest system called Cascade adds several explicit level to the hardware hierarchy as the maximum distance between any two nodes is 5 hops with the number of reachable nodes increasing by factors of 16, 6 and 240 after each hop, respectively [Faanes et al., 2012].

The following sections introduce the most important methods in HPC for increasing the speed of a computational model. With the exception of hierarchy-free methods discussed briefly in Section 3.5 every method pertains only to a specific level or aspect of the hardware hierarchy. For this reason most of the methods must be used in order to fully take advantage of supercomputing hardware.

3.4 Distributed memory parallelization

Distributed memory parallelization, also called distributed memory concurrency, means allowing an application to utilize several CPUs that cannot access each others' memory directly or transparently from the programming point of view. In other words the programmer must explicitly transfer data between the memory accessible by one CPU to the memory accessible by another CPU in order for the CPUs to be able to work in parallel. The defacto standard method for distributed memory parallelization in high performance computing is the message passing interface (MPI) specification [Message Passing Interface Forum, 2012] which provides an application programming interface for point-to-point and collective operations for sending and processing messages between processes. For this reason from hereinafter the term MPI program will be used to refer to a distributed memory parallel program.

Once a simple version of a program has been developed and it is clear that shorter run times are required (e.g. results are needed overnight instead of within weeks or more) the first step should be to parallelize the program using MPI. The most important reason that distributed memory parallelization should be the first consideration for speeding up a program is that the potential speedup factor can exceed 10 000 or even 100 000. Such a large speedup is quite impossible to achieve using any other method with the exception of an algorithm with better scaling behavior, which should have been investigated already when developing the serial program. There are also other advantages to designing/rewriting a program for

distributed memory parallelization as it tends to expose inherent scalability bottlenecks in the methods/algorithms used with respect to very large or computationally expensive systems. Removing these bottlenecks can also speed up a serial program significantly, although even a parallel program that runs 5 times slower than serially can be acceptable as it is already difficult to find new hardware with less than 6 or 8 computing cores. A parallel program developed with the aim of running on 100s or 1000s of cores should scale practically ideally to the dozen or so cores available in current shared memory hardware, already making it faster than the above serial program.

Naturally distributed memory parallelization is the only option if a simulation does not fit into one shared memory system or if the expected runtime is measured in days even when thousands of cores would be used efficiently. On the other hand in some fields, such as molecular dynamics, it is not always possible to increase the size of the system or the resolution in order to utilize a larger number of cores efficiently. For example simulating the behavior of a protein in water requires between 10 k and 100 k atoms [Hess et al., 2008] while scaling even a much larger system of 1 M atoms to 4 k cores is difficult [Sun et al., 2012]. In such cases the optimizations presented below can be more important than distributed memory parallelization but for all algorithms used in the work presented here this was not the case.

In practice, when implementing an algorithm using distributed memory parallelism, I have found that describing or transforming an algorithm to a local, i.e. cell-by-cell, representation is very helpful. Such a representation makes it immediately clear, or at least gives very strong indications as to, what kind of parallel scalability to expect. Moreover a cell-by-cell representation makes it clear what actually has to happen to the data of one cell and its neighbors in order to calculate the next time step in that cell. This also makes the planning of the programs's data structures and the required data transfers between processes during operation easy. For example GoL could be easily formulated using linear algebra but in my opinion that description would be unnecessarily complicated compared to the simple rules presented in Section 1.1, especially when considering AMR (Section 4.3). A more practical example follows in which a parallel solver for the Poisson's equation is developed on top of dccrg in order to implement removal of divergence of B in the parallel version of GUMICS.

3.4.1 Example: distributed memory Poisson solver

Poisson's equation $\nabla^2\phi = f$ relates the scalar potential represented by ϕ to, for example, a known distribution of mass or charge f (see the electric sail example in Section 2.1.3). The same equation also results when removing the divergence of the magnetic field [Brackbill and Barnes, 1980] by solving for ϕ from $\nabla^2\phi = -\nabla \cdot \mathbf{B}$ and setting the new magnetic field $\mathbf{B}_{new} = \mathbf{B} + \nabla\phi$ for which $\nabla \cdot \mathbf{B}_{new} = \nabla \cdot \mathbf{B} + \nabla^2\phi = 0$. Figure 3.1 shows the profound effect of divergence cleaning in a two-dimensional magnetospheric simulation in the XZ plane without an ionosphere when using an MHD solver that has not been designed to preserve the divergence of the magnetic field in two or more dimensions. The color coded pressure is shown after one simulated hour on a logarithmic scale, on the left without divergence cleaning

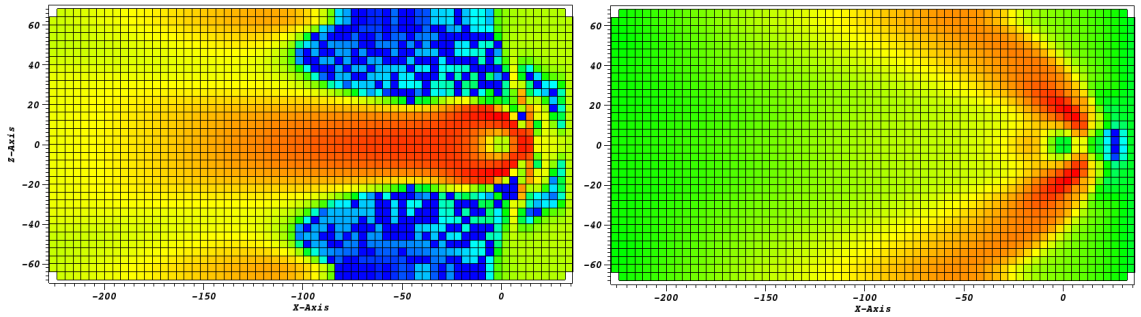

Figure 3.1: The effect of $\nabla \cdot \mathbf{B}$ cleaning on total plasma pressure in a two-dimensional magnetospheric simulation. No cleaning on the left, cleaning every 20 simulated seconds on the right, see the text for details.

and on the right with divergence cleaning every 20 simulated seconds. Without cleaning the minimum value is 10^{-15} Pa (minimum physical pressure, see Janhunen et al. [2012]) and the maximum is about $4 \cdot 10^{-11}$ Pa. With cleaning the minimum is about $4 \cdot 10^{-14}$ Pa and maximum about $8 \cdot 10^{-11}$ Pa.

A general numerical method for solving Poisson's equation, called the biconjugate gradient (BiCG) method, was selected because it supports non-symmetric equations which appear with AMR and because this method is also used in the serial version GUMICS. The use of an existing library for this purpose was considered but no suitable one was found. Either an otherwise suitable library was not parallel, e.g. Eigen[2] or it seemed that writing such a library on top of dccrg would have been easier than using existing parallel libraries / frameworks such as MUMPS[3] or Elmer[4].

An algorithm implementing the BiCG method is given in Section 2.7.6 of Numerical recipes [Press et al., 2007] in linear algebra form $\mathbf{A}\mathbf{x} = \mathbf{b}$, where the matrix $\mathbf{A}$ is a discrete version of ∇^2, $\mathbf{b} \equiv f(\mathbf{r})$ and $\mathbf{x} \equiv \phi(\mathbf{r})$ are vectors. The solution is obtained by first guessing the value of $\mathbf{x}_0$, which is always $\mathbf{x}_0 = 0$ in the divergence remover, and setting $\mathbf{r}_0 = \overline{\mathbf{r}}_0 = \mathbf{p}_0 = \overline{\mathbf{p}}_0 = \mathbf{b} - \mathbf{A} \cdot \mathbf{x}_0$. Subsequently, the following steps are performed for $k = 0..N$ until some norm of the residual $\mathbf{b} - \mathbf{A} \cdot \mathbf{x}_0$ is minimized or becomes small enough

$$\alpha_k = \frac{\overline{\mathbf{r}}_k \cdot \mathbf{r}_k}{\overline{\mathbf{p}}_k \cdot \mathbf{A} \cdot \mathbf{p}_k} \tag{3.1}$$

$$\mathbf{r}_{k+1} = \mathbf{r}_k - \alpha \mathbf{A} \cdot \mathbf{p}_k \tag{3.2}$$

$$\overline{\mathbf{r}}_{k+1} = \overline{\mathbf{r}}_k - \alpha \mathbf{A}^T \cdot \overline{\mathbf{p}}_k \tag{3.3}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \tag{3.4}$$

$$\beta_k = \frac{\overline{\mathbf{r}}_{k+1} \cdot \mathbf{r}_{k+1}}{\overline{\mathbf{r}}_k \cdot \mathbf{r}_k} \tag{3.5}$$

[2] http://eigen.tuxfamily.org/index.php?title=Main_Page
[3] http://graal.ens-lyon.fr/MUMPS/
[4] http://www.csc.fi/english/pages/elmer

$$\mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k \tag{3.6}$$

$$\overline{\mathbf{p}}_{k+1} = \overline{\mathbf{r}}_{k+1} + \beta_k \overline{\mathbf{p}}_k \tag{3.7}$$

which stores the solution to Poisson's equation in $\mathbf{x}_N$.

In order to implement this algorithm on top of dccrg it was reformulated into a "cell centric" form which makes the operations to be carried out for the data of each cell and its neighbors explicit. During iteration only the old and new values of the vectors are kept in memory. From hereinafter the subscript i in a variable will refer to the ith component of that variable, which is stored in cell i, instead of the whole variable at the ith iteration. Dot products of the form $\overline{\mathbf{r}} \cdot \mathbf{r}$ are then equal to $\sum_i \overline{r}_i r_i$ where $\overline{r}_i$ and r_i are stored in cell i and i iterates over all cells of the grid. The operation $\mathbf{A} \cdot \mathbf{p}$ is substituted by the discrete version of the ∇^2 (Laplace) operator applied separately in each dimension. In the following 1d analysis the dimension is denoted by x but when solving a problem in more than one dimension the same procedure is applied separately for each dimension by substituting y or z in place of x. The second spatial derivative is taken as the coefficient a in the equation for the parabola $f = ax^2 + bx + c$, which goes through the three points defined by the cell's and its neighbors' values on opposite sides in the x dimension. The coefficient can calculated from [Weisstein, 2013]

$$\begin{vmatrix} f^2 & x & f & 1 \\ f_{-x}^2 & x_- & f_{-x} & 1 \\ f_0^2 & x_0 & f_0 & 1 \\ f_{+x}^2 & x_+ & f_{+x} & 1 \end{vmatrix} = 0, \tag{3.8}$$

where f_0 is the value of f in the cell in which the seconds derivative is being calculated in and x_0 is its center's x coordinate, f_{-x} and f_{+x} are the values of f in neighboring cells in the negative and positive x directions from the cell respectively, and x_- and x_+ are the neighbors' locations. With the assumption that the cell itself is at the origin ($x_0 = 0$) this reduces the coefficient to

$$a = -\frac{x_-(f_0 - f_{+x}) + x_+(f_{-x} - f_0)}{x_- x_+ (x_+ - x_-)}. \tag{3.9}$$

which gives at the point $x = 0$, i.e. in the cell where the Laplace operator is being calculated

$$\frac{\partial^2 f}{\partial x^2} = 2a = 2\left(\frac{f_0}{x_- x_+} + \frac{f_{+x}}{x_+(x_+ - x_-)} - \frac{f_{-x}}{x_-(x_+ - x_-)}\right). \tag{3.10}$$

Thus the contribution from each neighbor j in x, y and z dimensions is $2f_j/(r_{offset} \triangle r)$ where r_{offset} is the distance between the cell and its neighbor (for x dimension either $x_- < 0$ or $x_+ > 0$) and $\triangle r = x_+ - x_- > 0$. With AMR, for example if a cell has four smaller face neighbors in positive x direction, f_{+x} is the average value of f in the neighbors, which assumes a cartesian grid. Furthermore, in the implementation the variables are scaled such that the diagonals of the matrix A are ones meaning that the final multiplier of f_0 is 1 instead of $2(x_- x_+)^{-1}$ and the neighbors' multipliers are adjusted accordingly.

In the discrete version of $\mathbf{A} \cdot \mathbf{p}$, used e.g. in Equation 3.2, the value $p_i + \sum_j c_{ij} p_j$ is calculated in each cell i, where j iterates over all face neighbors of the cell (in all dimensions). The multiplier of the component of p in cell i (p_i) is 1 due to scaling of the variables and the multiplier c_{ij} of p_j, which is stored in cell i, takes into account the effects of geometry ($2/(r_{offset} \triangle r)$, see above) as well as the number of dimensions since e.g. in two dimensions the final coefficient of f_0 without scaling is $4(x_- x_+ + y_- y_+)(x_- x_+ y_- y_+)^{-1}$. In the discrete version of $\mathbf{A}^T \cdot \mathbf{p}$, used in Equation 3.3, the same steps are executed as in $\mathbf{A} \cdot \mathbf{p}$ except that the coefficient c_{ji}, which is stored in cell j, is used instead of c_{ij} for updating the value of cell i from its neighbor j. This is the exact opposite of what was done in $\mathbf{A} \cdot \mathbf{p}$. In an MPI program local copies of the values of p, in cells that are located on other processes, must be updated before carrying out the required calculations in local cells (that have neighbors on other processes).

The parallelization of the above algorithm using dccrg was not difficult after the serial version was written and tested, although future parallelization was taken into account from the beginning in order to ease the eventual parallelization effort. For example in the main solver function [Honkonen, 2013] only 6 out of more than 200 lines of code deal with distributed memory parallelization. Of those 6 lines two are for setup and are executed only once during the algorithm. Two lines update the required remote neighbor data and two obtain and distribute the results of the algorithm's two dot products between all processes during iteration. Most of the difficulties in development were not related to parallelization but due to, for example, supporting AMR or a stretched cartesian grid and scaling the diagonal of the matrix A to 1.

3.5 Hierarchy free methods

Hierarchy free methods, algorithms and implementations are constructed in such a way that the existence of one or more hardware hierarchies does not hinder the programs' performance significantly relative to a program optimized for a specific hardware configuration. Cache-oblivious algorithm is a common name for these type of methods as they are used for improving the cache-efficiency of algorithms. Traditional cache-aware optimizations include e.g. the usage of blocks of cells [e.g. Stout et al., 1997] whereby the data of all cells within a block is contiguous in memory.

In its simplest form a cache-oblivious algorithm changes the order in which data are stored in memory or the order in which they are processed so that during processing only "nearby" items are ever needed which most likely already exist in the cache. A common strategy in cache-oblivious algorithms is to use recursion to divide the work into smaller and smaller localized problems. Once a particular subproblem fits into the cache it can be solved without additional cache misses. See Frigo et al. [1999], Prokop [1999] for details and examples of cache-oblivious algorithms for matrix-matrix multiplication, fast Fourier transform and sorting. Heinecke and Trinitis [2012] show that cache-oblivious algorithms are also suitable for hardware with wide SIMD capabilities and a large number of cores (see Sections 3.6 and 3.7 respectively).

The data locality optimization achieved with a cache-oblivious algorithm is also applicable to a distributed memory system, especially as the number of hardware hierarchies increases. By optimizing the location of data in the system the total network throughput available to a simulation can be increased significantly [e.g. von Alfthan et al., 2013a].

3.6 Vectorization

In bitwise operations such as XOR the bits are independent of each other and hence these hardware operations are easy to extend to larger integer types. In arithmetic operations which are more useful for physical simulations this is not the case and hence specialized hardware is required in order to be able to perform a calculation such as multiplication of several variables in parallel. This act of performing a single instruction with multiple data (SIMD) will be from hereinafter referred to as vectorization.

Starting from 1960s and up until the late 1990s vectorization was the most important method of utilizing the full computational capacity of supercomputers. After the performance/price ratio of commodity CPUs without vector instructions increased well above that of vectorized hardware, and was consequently used also in supercomputers, vectorization became less important for obtaining good performance. For about 10 years vectorization did not improve the performance of HPC programs significantly but with the introduction of the Advanced Vector Extensions (AVX) 1 and 2 [Buxton, 2011] vectorization could again improve application performance by over an order of magnitude. For example by taking advantage of vectorization explicitly in the PIC electric sail model [further developed from Janhunen and Sandroos, 2007] by using a specific C++ vector class library [Fog, 2013] the number of propagated particles per second increased by a factor of 8 to 56 % of peak theoretical floating point performance on a 2.6 GHz Intel Core i5 2540M CPU (P. Janhunen, private communication). While there is no standard way of explicitly taking advantage of vectorization in C, C++ or Fortran, libraries can make the problem easier to tackle or even almost completely transparent to the programmer [Fog, 2013].

3.7 Threading

Around 2001 both Intel and AMD [Advanced Micro Devices, 2001] started producing desktop CPUs with multiple cores, i.e. independent units executing CPU instructions in parallel, that could access the same main memory. Thus it was possible for a program to consist of several threads that were executed in parallel thereby multiplying the amount of computational work done by the number of CPU cores. Access to the main memory is also largely transparent from the programming point of view. Hence threaded numerical models are relatively easy to develop, especially with high-level libraries such as OpenMP.

A distributed memory program cannot access the memory of other nodes and so needs to keep a up-to-date copy of the required data in local memory. In a

threaded program this is not required which saves memory and also takes better advantage of the CPU's caches. The speedup factor of a threaded program relative to a MPI parallel program is usually in the range 1...3 depending on factors such as computation/communication ratio, node interconnect performance, number of cores per CPU, CPUs per shared memory node, number of simulation cells per node, etc. For many problems a larger speedup for an MPI program might be obtained by taking advantage of vectorization instead of threading.

3.8 Graphics processing units

As the popularity of more and more realistic looking 3d video games increased at the turn of the millenia so did the performance and programmability of graphics cards (GPU) which handle the calculations necessary for rendering the games' content onto the screen. While initially GPUs could only render pixels to the screen through a fixed set of steps eventually GPU manufacturers started to make the hardware (more) programmable and exposed that functionality to users. Even at a point where GPUs were quite restrictive in features [Thompson et al., 2002], for example there were only 21 instructions and the maximum program length was 128 instructions, they could already outperform CPUs in matrix multiplication and similar problems by a significant factor (up to 3...10 depending on the problem) provided that the size of the problem was large enough.

What separates GPUs from other performance enhancement techniques, at least at the moment, is the requirement to write the code running on them using a separate language which is typically either OpenCL or CUDA. As the number of libraries providing easy access to GPU computing increases [e.g. ViennaCL Rupp et al., 2010] and their quality improves it is becoming possible to easily take advantage of GPUs even in interpreted language environments such as Python [Kloeckner, 2012]. But similarly to the floating point unit of modern CPUs which was originally a separate and optional chip [Intel, 1989] CPU/GPU manufacturers seem to be integrating the functionality of the GPU back into the CPU. An example of this is the Xeon Phi brand of co processors [Intel, 2012] already in production which consist of up to 61 cores ($\times 4$ threads) each with a 512 bit wide vector register. Taking advantage of this large number of cores in parallel does not require another language but only threads (POSIX, OpenMP, C++11, etc.), thus easing the software development effort and also allowing programs with complex algorithms and data structures to utilize the whole computational capacity of the system. On the other hand Xeon Phi will further increase the hierarchical nature of hardware with even wider vector registers and by making the L2 cache of each core connect to only a subset of all L2 caches over a 1d interprocessor ring network.

3.9 Rewriting a program in a faster language

A speedup of up to a factor of 10 can be expected by moving from an interpreted language such as Python to a compiled language such as C. On the other hand for standard operations such as linear algebra an optimized and compiled library is likely

already used by the interpreted language possibly leading to a less than a factor of two overall speedup. In practice it could be much more useful to rewrite only the low level time consuming parts of the application in a faster language and keep the high level control and program logic in a language with more features and better readability. One example of a widely used program following this method is GPAW [Enkovaara et al., 2010] which is used for electronic structure calculations using time-dependent density-functional theory. In case the development objective from the start is a massively parallel program requiring the largest available resources, starting development directly in a compiled language is also justified, for example in order to reduce overheads especially related to memory management.

> It is only ones and zeros, it cannot be hard
>
> Unknown

Chapter 4

The DCCRG parallel grid library

As discussed in Section 1.1, the basic structure of many numerical models is identical from the software development point of view. The simulated volume is discretized and at each discrete point the solution on the next time step or iteration depends on data within a finite distance from that point. In a distributed memory machine (part of) the data may need to be transferred from the memory of one process to the memory of another process. With the number of CPU cores in supercomputers increasing exponentially it is becoming increasingly difficult to utilize all CPU cores efficiently. In Article I a library (dccrg) was developed which abstracts away many details related to developing a numerical model with good parallel scalability and advanced features such as run-time adaptive mesh refinement (AMR). As an example of the usage of dccrg Figure 4.1 shows a complete parallel program playing Conway's Game of Life (GoL), updated from Figure 4 of Article I. In the current version of dccrg cells return information about data transfers in the same format as used by MPI functions themselves (lines 11-15), which allows for greater flexibility than previously. An instance of dccrg is created on line 26 and initialized on lines 27-28. As in the version of dccrg presented in Article I, the only dccrg function call required in a parallel program is on line 40 which updates cell data between processes.

4.1 Parallelization details

The fundamental aspects of the serial implementation of dccrg are presented in Article I (e.g. Figures 2 and 3), e.g. how cells are stored in memory and the algorithm for searching for cells' neighbors. In a parallel setting each cell is owned by one process and the cells whose data must be exchanged between processes are defined by the neighbors of each cell. The neighbors of a cell are defined by its neighborhood, which is identical for all cells. In the current version of dccrg using one or more nearly arbitrary neighborhoods is supported, the only limitation being that the largest distance of neighbors from a cell cannot exceed the initial value of the neighborhood size. An example is shown in Figure 4.2 where each cell's data is sent to the owner(s) of the cell above of and to the right of the cell. Given a grid whose cells are distributed among three processes and which is periodic in the horizontal direction, the bottom of Figure 4.2 shows which cells' data is sent to and from process 1. Notice the asymmetry of data transfers between processes 1 and 3 as the neighborhood of

each cell is defined in units of the size of the cell itself. Due to this, cells of process 3 do not consider cells of process 1 as neighbors and hence no data is sent from process 1 to 3.

One of the most important features of dccrg is the ability to easily send the required cell data between processes, in the best case, by calling only one function (e.g. line 40 in Figure 4.1). Figure 4.3 shows how this is achieved internally in the case of Figure 4.2 between processes 1 and 3. The `get_mpi_datatype` method of the user's cell data class, which is stored in each grid cell, defines the data to be transferred to and from that cell. This method is used by dccrg to query the actual data to be transferred between processes as defined by the cells' neighbors. Note that the data represented by each cell's datatype, shown in Figure 4.3 with pink ellipses for two cells' datatypes, does not have to be contiguous in memory. From the datatypes of cells to be sent / received a final datatype is constructed which is subsequently given to `MPI_Isend` or `MPI_Irecv`. This allows for great flexibility in the type of data that can be transferred between processes as dccrg does not have to care about the structure of the data but just passes on the information to MPI library functions. The example in Figure 5 of Article I shows how this approach can be used in a numerical model in which the amount of data in each cell varies as a function of both space and time. In principle, the above approach to data transfers is optimal in the sence that data from local memory of one process is transferred directly to the local memory of another process without creating additional copies, but in practice this depends on the hardware and the MPI implementation.

4.2 Notable new features

Several features unique to dccrg are detailed in Article I and since then more features have been added. In numerical models where the amount of data varies on a cell-by-cell basis the computational load must also be balanced in multiple steps which is now supported by dccrg. For example the latest version of Vlasiator [von Alfthan et al., 2013b] uses a sparse representation of the distribution function where only those velocity blocks exist in which the value of the distribution function is above a user-defined threshold. When sending a cell to another process, the amount of velocity blocks being transferred is not known in advance by the receiving process, hence the number of velocity blocks must be sent first in order to reserve memory for incoming velocity block data on the receiving process. Using several rounds of communication also allows Vlasiator to send fewer cells at a time between processes reducing the memory requirements of the simulation. When a fraction of cells have been transferred from a process the memory used by their data can be reallocated for incoming cell data.

The method of transferring data between processes using MPI datatypes shown in Figure 4.3 is also used by dccrg to provide efficient and easy to use functions for parallel file I/O using MPI[1]. When saving grid data into a file dccrg queries the MPI datatypes of all local cells and constructs a final datatype representing all the cell data to be saved. Additionally a contiguous datatype for the same data

[1]Thanks to S. von Alfthan for this simple yet powerful idea

is created for "receiving" the data into a file. At this point each process scatters the number of bytes it will write to all other processes so that each process can calculate the file offset where to start writing its data. The data written by each process consists of local cell ids, the offsets of their data in the file and the cell data itself given by the user's cell data class. These are written using a total of two calls to the `MPI_File_write_at_all` function. Currently the geometry of the grid, i.e. the starting coordinate of the grid and the physical location, size and shape of cells, must be saved and loaded by the user but adding support for doing that automatically in dccrg is planned.

Another feature added to dccrg after the publication of Article I is the ability to construct a new instance of dccrg based on another, already initialized instance, but using a different cell data class. This feature is used in the parallel solver of Poisson's equation implemented on top of dccrg which is presented in Section 3.4.1. The feature allows the solution to the Poisson's equation to be calculated independently from any particular numerical model.

4.3 Testing dccrg

There are at least two popular methods or categories of software development. In test driven development [Beck, 2002] a test for a desired feature is written before the feature itself is implemented and iteratively improved. In feature driven development [Palmer and Felsing, 2002] a feature is planned, developed and tested in approximately this order. The development of the dccrg was mostly driven by desired features, initially by those required for parallelizing the GUMICS model and subsequently by the features required by Vlasiator. The most important features of dccrg include transparent (usually requiring only one line of code) updates of cell data between processes whose cells are neighbors, arbitrary data in grid cells i.e. data whose size can change at run-time and independently in each cell, AMR and ease of use.

Different implementations of GoL have been exceptionally helpful for testing the various algorithms of dccrg. Perhaps most importantly GoL has many features in common with much more complex numerical models while still being extremely simple. This has helped to design a simple as possible application programming interface for dccrg that also retains the functionality required for more complex models. On the other hand the discrete nature of GoL and its complex, almost chaotic, behavior made any mistakes in the algorithms of dccrg readily apparent, such as bugs in AMR, remote neighbor updates or user-defined neighborhoods[2].

In order to test AMR using GoL the solver was extended to a grid with cells of different size in such a way that the overall result remains identical to the basic game. Figure 4.4 shows an example from the AMR GoL test of dccrg where a glider pattern is calculated in a grid where random cells are refined and recoarsened every other time step respectively. The computational load is also "balanced" in the test by transferring cells to random processes at each time step and is shown in Figure 4.5 for a run using 3 processes. The life state of siblings, i.e. all children of

[2] i.e. stencils

the same refined cell, is always equal and they behave identically to their parent. Their effect on other cells is also identical to the effect their parent would have. While this applies also recursively, i.e. cells' can be refined more than once, it is not a completely generic test for AMR, but it was used nevertheless in order to take advantage of the large number of existing GoL patterns designed for a non-adapted grid.

When calculating the number of live neighbors a large cell can check the state of any one sibling of a refined cell without additional communication. On the other hand a small cell must inform its siblings of the number of its live neighbors that are not neighbors of its siblings. In general this is achieved by $2^L - 1$ rounds of extra updates of remote neighbor data where L is the maximum refinement level of cells that exist in the grid. The information exchanged consist of the state of the siblings' own (non-sibling) neighbors. Each round of communication informs one more layer of siblings about the life state of a small cell's neighbors. After L rounds of communication all siblings know the life state of all cells that would be neighbors to the siblings' (grand...)parent and the GoL algorithm can proceed in a standard way. The current version of the AMR GoL test in dccrg only supports a maximum refinement level of 1 for cells (largest possible cells have refinement level 0).

The ability to use additional almost arbitrary neighborhoods (see Section 4.1) for cells is also tested using, for example, a GoL program. Figure 4.6 shows a game in which a cell's neighborhood consists of eight neighbors (O) which are a distance of one cell further away than normal from the cell (X). This allows four interlaced games to be played on the same two-dimensional grid as cells located at an even index (see Figure 2 in Article I) do not affect cells located at an odd index in either dimension of the grid. This is shown in Figure 4.6 by coloring cells located at indices $(2i, 2j)$, where $i, j = 0...N - 1$, starting from the upper left as yellow, cells at $(2i + 1, 2j)$ as red, at $(2i, 2j + 1)$ as cyan and at $(2i + 1, 2j + 1)$ as green. For example the yellow glider correctly moves through the simulation unaffected by cells of other colors. Using a different neighborhood for cells (without AMR) does not require modifications to the GoL solver and e.g. the one shown in Figure 4 of Article I works correctly as is.

```cpp
#include "boost/array.hpp"
#include "boost/mpi.hpp"
#include "boost/tuple/tuple.hpp"
#include "vector"
#include "zoltan.h"
#include "dccrg.hpp"

struct game_of_life_cell {
   int is_alive = -1, live_neighbors = -1;

   boost::tuple<void*, int, MPI_Datatype> get_mpi_datatype(
      const uint64_t, const int, const int, const bool
   ) {
      return boost::make_tuple(&(this->is_alive), 1, MPI_INT);
   }
};

int main(int argc, char* argv[])
{
   boost::mpi::environment env(argc, argv);
   boost::mpi::communicator comm;

   float zoltan_version;
   Zoltan_Initialize(argc, argv, &zoltan_version);

   dccrg::Dccrg<game_of_life_cell> grid;
   const boost::array<uint64_t, 3> grid_length = {10, 10, 1};
   grid.initialize(grid_length, comm, "RCB", 1, 0);
   grid.balance_load();
   const std::vector<uint64_t> cells = grid.get_cells();

   // initial state
   for (auto cell: cells) {
      game_of_life_cell* cell_data = grid[cell];
      cell_data->is_alive = cell_data->live_neighbors = 0;
      if (cell % 4 == 0) cell_data->is_alive = 1;
   }

   for (int turn = 0; turn < 10; turn++) {
      grid.update_copies_of_remote_neighbors();

      // collect live neighbor counts
      for (auto cell: cells) {
         game_of_life_cell* cell_data = grid[cell];
         cell_data->live_neighbors = 0;

         const std::vector<uint64_t>* neighbors = grid.get_neighbors_of(cell);
         for (auto neighbor: *neighbors) {
            if (neighbor == dccrg::error_cell) continue;
            game_of_life_cell* neighbor_data = grid[neighbor];
            if (neighbor_data->is_alive == 1) cell_data->live_neighbors++;
         }
      }
      // assign new state
      for (auto cell: cells) {
         game_of_life_cell* cell_data = grid[cell];
         if (cell_data->live_neighbors == 3) cell_data->is_alive = 1;
         else if (cell_data->live_neighbors != 2) cell_data->is_alive = 0;
      }
   }
   return 0;
}
```

Figure 4.1: A complete parallel program playing Conway's Game of Life implemented with dccrg. Updated from Figure 4 of Article I.

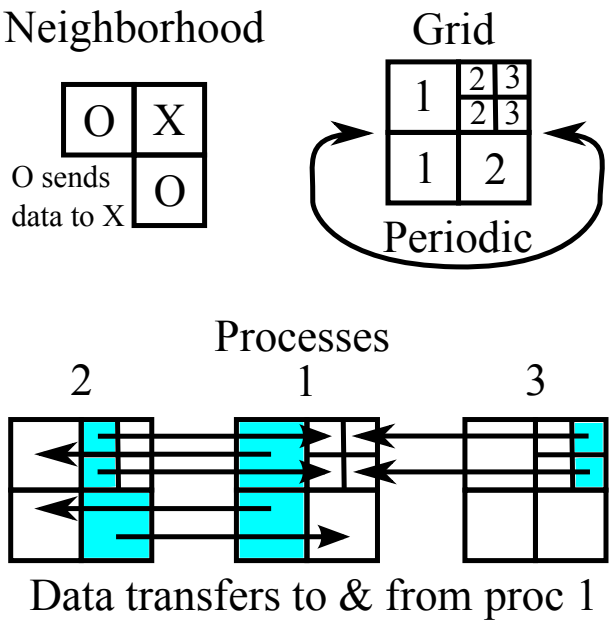

Figure 4.2: Data transfers to and from process 1 in the grid shown at the upper right with the cell neighborhood shown at the upper left.

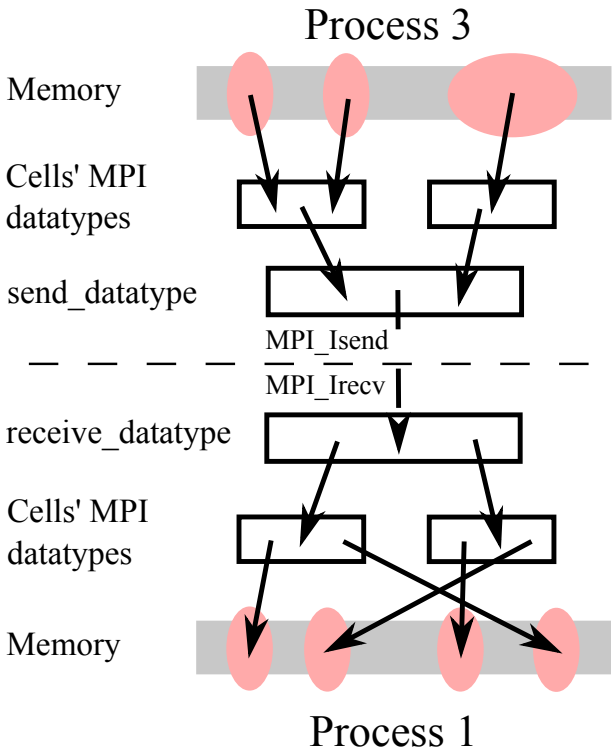

Figure 4.3: Schematic of how dccrg sends cell data from process 3 to process 1 in Figure 4.2.

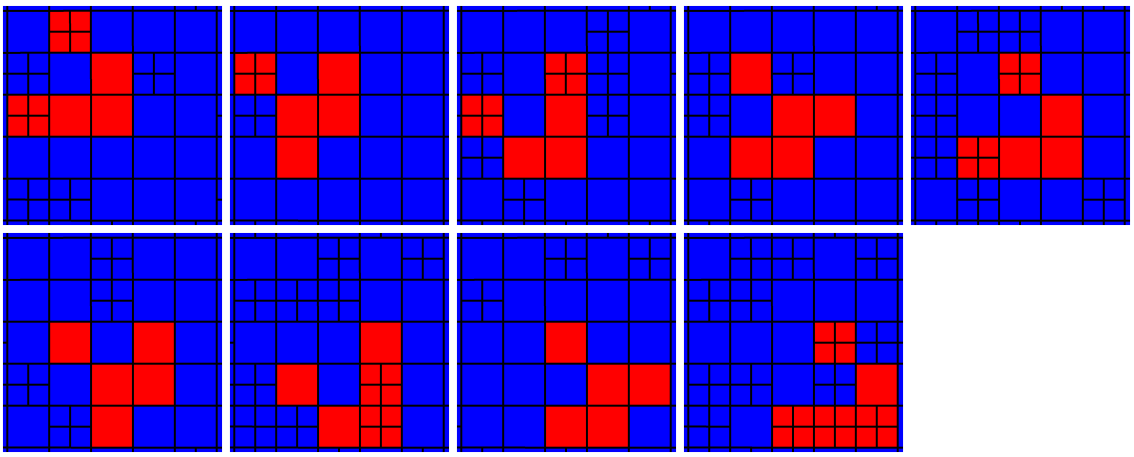

Figure 4.4: Conway's Game of Life played on a run-time adaptive grid. Red cells are alive, blue cell are dead. Nine time steps are shown in the order left to right, top to bottom.

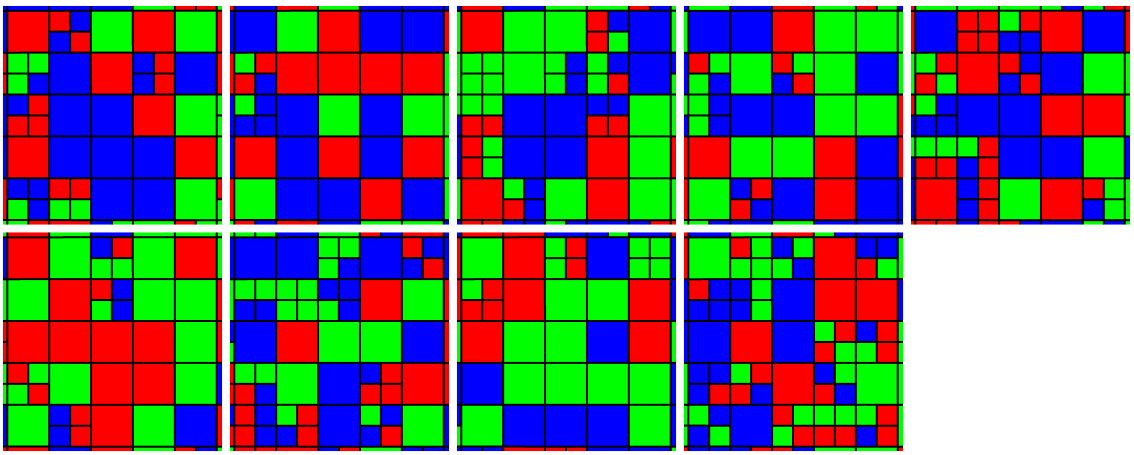

Figure 4.5: Same as Figure 4.4 but showing color coded the owner (MPI process number) of each cell.

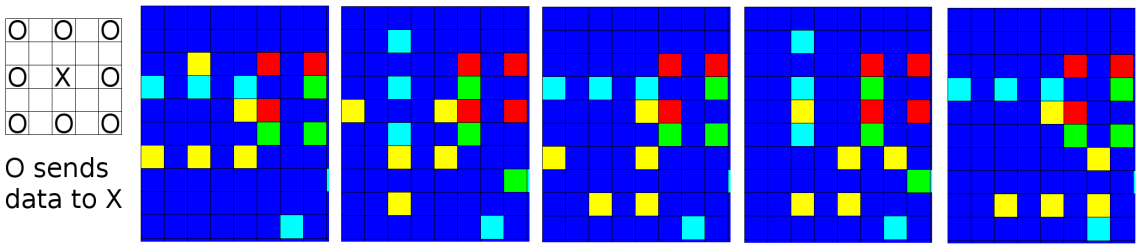

Figure 4.6: Four interlaced Games of Life played in the same grid. The neighborhood shown on the left is used for all cells in DCCRG, see the text for details.

Chapter 5

Verification and validation

> "regression testing"? What's that? If it compiles, it is good; if it boots up, it is perfect.
>
> Linus Torvalds on releasing a new version of Linux

In principle, or from a philosophical point of view, neither a hypothesis nor a numerical model can be proven correct, instead they can only be shown to be probable [Oreskes et al., 1994, and references therein]. Correctness is also distinct from verification which in turn means, strictly speaking, that the program has been formally proven to be true [as done e.g. in Harrison, 2003]. Furthermore, the term valid can be applied to a program in general but is misleading at best if applied to any specific model result(s) [Oreskes et al., 1994]. While the above definitions can lead to the notion that numerical models are only useful for guiding further study and are used only because of lack of access to the phenomena of interest, in practice numerical models have proven (informally) to be immensely useful for example for weather and space weather prediction. Even though terms such as confirmation or corroboration better describe what is done in science, due to the widespread use of the terms verification and validation (V&V) the latter will also be used here [Babuska and Oden, 2004].

The terms testing, verification and validation each have a specific meaning that overlap only to a small extent. An extensive treatment of V&V is given by Oberkampf and Trucano [2002] while specifics relating to space plasma are discussed by Ledvina et al. [2008]. Basically verification quantifies how well a numerical model solves its equations while validation quantifies how well the model's results correspond to reality. In this thesis testing will be used to refer to assessing the basic functionality of a program or a library and the V&V of relatively simple algorithms compared to e.g. global MHD modeling of the Earth's magnetosphere-ionosphere system.

5.1 Overview of GUMICS and Vlasiator

The Grand Unified Magnetosphere Ionosphere Coupling Simulation (GUMICS) models the Earth's entire magnetosphere, its interaction with the solar wind and its coupling to the ionosphere [Janhunen et al., 2012]. The magnetosphere is modeled using the conservative form of ideal MHD equations (Equations 2.1 without Q, 2.13, 2.14, 2.12 and 2.15) and it is coupled to an electrostatic height-integrated, i.e. 2d, ionosphere. The two way coupling is accomplished by mapping the field aligned currents from the inner boundary of the magnetosphere into the ionosphere and by mapping the ionospheric potential back into the inner boundary of the magnetosphere along the Earth's intrinsic dipole field. The field aligned currents are calculated from the magnetic field and the potential is solved from the current continuity equation in the ionosphere. Further details are provided in Janhunen et al. [2012] and Article III.

Vlasiator is a space plasma model based on the Vlasov description of a collisionless plasma (Eq. 2.4) that is being developed at the Finnish Meteorological Institute in the QuESpace project. Vlasiator is a hybrid Vlasov model as it only solves the motion of ions while the evolution of the magnetic field is governed by Eq. 2.12, i.e. electrons are considered as a charge neutralizing massless fluid. See [Palmroth et al., 2013, von Alfthan et al., 2013b] for additional details.

5.2 Verifying Vlasiator

The basic question that verification tries to answer is does the numerical model solve the equations correctly [Ledvina et al., 2008]? Ideally verification would provide proof that the algorithms implemented by the code approximate the equations correctly and that the algorithms converge to the correct solution [Oberkampf and Trucano, 2002]. As it is unlikely that such proof will ever exist for computational fluid dynamics programs, verification then becomes the absence of proof that the program is incorrect. The amount, or lack, of such evidence can be especially important when the software is considered for use by other scientists.

When verifying a numerical model it can be compared to analytic solutions, obtained e.g. using the method of manufactured solutions [as e.g. in Welling et al., 2012], and/or if that is not possible to other numerical models. A classic, although not necessarily good, example of verification of space plasma models is the GEM magnetic reconnection challenge [Birn et al., 2001, and references therein] in which the reconnection rate in numerical models from resistive MHD to PIC are compared. In case of global modeling of the Earth's magnetosphere, i.e. the time-dependent interaction of the solar wind, magnetosphere and ionosphere, only comparisons to other numeric/empirical models are feasible.

Initial verification of Vlasiator consisted of various tests for the solution of the 6d advection equation. In these tests Vlasiator was not yet self-consistent i.e. B was not solved as described in Section 2.1.4 but was set externally along with E. This approach is essentially identical to the test particle method in which charged particles are propagated in externally set EM fields (e.g. protons in Moore et al. [2005] and electrons in Ashour-Abdalla et al. [2011]). The largest test particle simulation

using Vlasiator to date was carried out in Article II where the behavior of plasma was studied in a global magnetospheric simulation. The EM fields were taken from the GUMICS results of Article IV and updated in Vlasiator after every simulated minute. The computational demands of this test have been estimated in Honkonen et al. [2011] and are many orders of magnitude higher than e.g. in typical GUMICS simulations. A 5d version of this test with 2 spatial and 3 velocity dimensions was conducted using GPUs in Sandroos et al. [2013] and showed that Vlasiator can utilize multi-GPU systems efficiently at least for solving local problems.

The above test particle tests indicate that the FVM solver implemented in Vlasiator works correctly also in a global magnetospheric simulation. As the EM fields in these tests are taken from the GUMICS solution it is expected that the overall Vlasiator solution significantly resembles that of GUMICS. The solar wind boundary condition in Vlasiator is correct as shown by the density and velocity trajectory plots in Figures 1, 2 and 3 of Article II. Since the inner boundary of the Vlasiator simulation is around 7 R_E and the ionospheric outflow present in GUMICS is absent in Vlasiator it does not make much sense to compare the two simulations closer than about 10 R_E from Earth. Plasma velocity in Vlasiator seems to agree with GUMICS better than plasma density. This is probably due to the very low spatial resolution of 1 R_E in Vlasiator compared to the 0.25 R_E resolution of GUMICS and adequate velocity space resolution in Vlasiator for capturing MHD flow features. It is also possible that the velocity space resolution is sufficient for the kinetic effects of the plasma to be present as shown by the north-south asymmetry of the dayside flow field in Vlasiator compared to GUMICS in Figure 1a of Article II. The effects of different spatial and velocity resolution in local self-consistent hybrid-Vlasov simulations are investigated in Kempf [2012] and in global 5d simulations of the Earth's magnetosphere in von Alfthan et al. [2013b]. The local tests showed that both too low ordinary space and velocity space resolutions can produce noticeable artifacts in the solution. The global tests showed that the lowest acceptable velocity space resolution for modeling the Earth's magnetosphere is about 30 km s^{-1} despite the reasonable results obtained in Article II using 50 km s^{-1} resolution.

An obvious difference between the Vlasiator test particle simulation and GUMICS is the plasma velocity distribution, for example at the sunward magnetopause. While the bulk plasma velocities in both models slightly below the Sun-Earth line in Figures 2 a and b of Article II are much less than the solar wind speed, the plasma in Vlasiator is actually gyrating around the magnetic field with an average velocity of about 300 km/s with respect to the bulk value as shown in Figure 5.1. Such ion ring distributions in velocity space can create, for example, shocklets observed upstream of the Earth's bow shock [Killen et al., 1995]. These effects would be nearly impossible to model using MHD theory, even with multiple fluids and pressure anisotropy.

5.3 Validating GUMICS and other MHD models

The basic question that validation tries to answer is does the model represent nature [Ledvina et al., 2008]? Prior to validation a model must be verified properly otherwise errors from different sources might cancel each other out [Oberkampf and

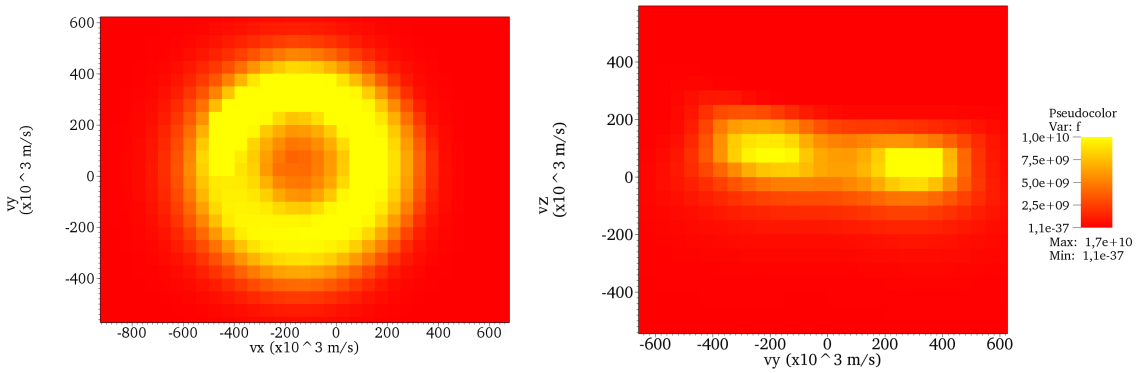

Figure 5.1: Velocity distribution of plasma at the magnetopause on the Sun-Earth line in a global magnetospheric test particle simulation [from Honkonen et al., 2011].

Trucano, 2002] leading to a false positive result. Without verification it is also difficult or impossible to attribute differences between model results and observations to errors in programming (software bugs), numerics (resolution, algorithms) and physics (analytical model). Also validation should not be confused with calibration where the parameters of a properly verified model are adjusted in order to minimize the final error over a range of experiments [Oberkampf and Trucano, 2002], either manually or automatically [as done e.g. in Laine et al., 2012].

Validation of global magnetospheric models is especially difficult as the number of point measurements in the magnetosphere is limited at best. Until recently the validation of global models such as GUMICS has mostly consisted of studies of single events and qualitative comparisons to observations. Examples include the magnetic field observed by a spacecraft (Article IV), the polar cap boundary observed by an incoherent scatter radar Aikio et al. [2013], the cross polar cap potential calculated from high frequency radar observations of ionospheric flow [used in Article III, see Chisham et al., 2007, and references therein for the method itself] and indirect observations of energy transfer through the magnetopause [Palmroth et al., 2012]. More comprehensive, but still qualitative, validation was performed by Anekallu et al. [2013] who investigated energy conversion at the magnetopause based on 8 years of spacecraft observations of magnetopause crossings and a comparison to GUMICS.

Extensive quantitative validation of GUMICS against empirical models has not been conducted until recently. In this context an empirical model is a time-independent mathematical model that gives the average value of a parameter during specific solar wind conditions based on a large collection of observations, for example data from 11 different spacecraft over a period of 20 years [Fairfield and Jones, 1996]. In Figure 4 of Janhunen et al. [2012] the position of the last closed magnetic field line in the Sunward direction from the Earth, called the standoff distance from hereinafter, from the GUMICS results of 16 event simulations are compared against the empirical magnetopause models of Shue et al. [1998] and Lin et al. [2010]. For GUMICS runs in which the standoff distance is above about 10 R_E the results are reasonable as the GUMICS standoff distances fall between the empirical models. For GUMICS

runs in which the standoff distance is between 6 and 9 R_E it seems to have been underestimated as the corresponding value in empirical models is usually higher than in GUMICS. In Article III the standoff distance for one event in four different global MHD models was investigated. In this particular case the empirical standoff distance varies between 8 and 11 R_E but GUMICS-4 seems to overestimate the standoff distance on average by about 1 R_E for unknown reasons. Standoff distance in the LFM model [Lyon et al., 2004] seems to follow the empirical estimate well which could be due to e.g. differences in the inner boundary condition or LFM using the highest order spatial discretization of the MHD equations.

Figure 5 of Janhunen et al. [2012] shows the correlation between B_y in the solar wind and in the near-Earth magnetotail in GUMICS based on results from 17 event simulations. Based on a linear fit the average penetration of B_y from the solar wind into the magnetotail in GUMICS is about 50 % while the average value based on observations is 60 % [Sergeev, 1987].

Gordeev et al. [2013] used 162 synthetic simulations with a constant solar wind to validate the GUMICS model against empirical models for the magnetopause, tail magnetic field, plasma sheet pressure, neutral sheet and cross polar cap potential of the ionosphere. GUMICS seems to reproduce the shape of the magnetopause well along with the shape of the neutral sheet while the plasma sheet pressure is slightly underestimated. On the other hand the magnetic field strength in the magnetotail is underestimated by 20 - 50 % in GUMICS perhaps due to lower reconnection rate than in reality. This is a well known shortcoming of ideal MHD which does not admit reconnection [e.g. Koskinen, 2011], hence reconnection in GUMICS is due to numerical diffusion.

Systematic and quantitative validation of global models using identical input has not been undertaken until recently with the works of Pulkkinen et al. [2011] and Rastätter et al. [2011]. In both cases various metrics were used to objectively quantify the performance of different empirical and numerical magnetospheric models by comparing to observations from ground magnetometers and geosynchronous spacecraft respectively. In Article III, for the first time to my knowledge, the performance of four different global models is quantified by comparisons to observations in the outer magnetosphere and the ionosphere. Important conclusions from Article III include: 1) magnetic field predictions of numerical models in the far tail are worse than using an average value with the exception of B_z, 2) increasing model resolution or coupling of additional physics does not automatically increase model performance.

As a final remark I have to note the recent conclusion of the "year run" conducted by G. Facskó et al. in which an entire year was simulated using GUMICS based on solar wind data obtained from NASA's OMNIWeb service. The simulation results should provide an excellent opportunity for validation of GUMICS on an unprecedented scale. The need for statistical studies of model performance was mentioned in Section 6.6 of Article III and the year run should provide significant insight into the physical performance of GUMICS.

Chapter 6

Advancing science by using a global MHD simulation

The purpose of scientific software is of course to do science. Global magnetospheric MHD models provide many possibilities for scientific discovery in parallel with in-situ observations and theoretical studies. In this chapter one such example is given based on the work carried out in Article IV where a new hypothesis was formulated from the results of the global MHD model GUMICS. Forming the same hypothesis based on observational data would have been much more difficult as it would probably have required an exceptional amount of observations in the Earth's magnetotail as well as near the cusps.

6.1 Introduction to plasmoids

Plasmoids that form in the Earth's magnetotail are large magnetic structures that remove plasma and energy deposited into the magnetosphere by the solar wind [Hones, 1979]. Historically the structure of the magnetic field in the magnetotail was considered only in 2d leading to the notion that plasmoids are closed loops of magnetic field propagating downstream. As observations contradicting the 2d view of plasmoids started to emerge [Hones et al., 1982] 3d models of plasmoid formation were developed [Hughes and Sibeck, 1987]. The 2d and 3d views of a plasmoid is shown in Figures 1 and 2 of Article IV respectively. From the point of view of the magnetic field the structure of the magnetosphere can be divided into three areas in which the magnetic field is connected to the Earth: 1) at both ends, 2) at only one end and 3) is not connected to the Earth. This structure changes due to magnetic reconnection which converts magnetic energy to kinetic energy by accelerating ions and electrons. There are several ways of defining a plasmoid in 3d, see for example Section 4.1 in Article IV. The definition used in Article IV is advantageous especially for automatic detection of plasmoids as it does not depend on arbitrary parameters possibly selected on a case by case basis.

Reconnection in the magnetotail can be divided into two categories: driven, a.k.a. forced, reconnection and non driven, a.k.a. spontaneous, reconnection [e.g. Lee et al., 1985, and references therein]. In driven reconnection the pileup of magnetic flux into the tail lobes is traditionally considered to be the main cause of reconnection, leading

to e.g. plasmoid formation. In Article IV a new mechanism for plasmoid formation was proposed in which large ($\approx 180°$) and fast ($\approx 10°$ min^{-1}) rotation of the IMF in the plane perpendicular to the Sun-Earth line leads to plasmoid formation.

6.2 Plasmoid formation in GUMICS

Based on a GUMICS simulation of a multiple substorm event it was argued in Article IV that large ($\approx 180°$) and fast ($\approx 10°$ min^{-1}) rotations of the IMF lead to the formation of plasmoids in the Earth's magnetotail. This was further supported by the results of Article III where a plasmoid with very similar structure of the magnetic field topology was formed also in the BATS-R-US model and a plasmoid also seemed to form in the LFM model. In both cases where a plasmoid formed in GUMICS in Article IV the IMF clock angle, when looking at Earth from the Sun, first rotated about 180 degrees clockwise from approximately -120 degrees to 40 degrees and after about 20 min rotated 180 degrees counter clockwise. The clock angle is defined as 0 degrees when IMF is in the $+Z_{GSE}$ direction, 90 degrees when in $+Y_{GSE}$ and -90 when in $-Y_{GSE}$ direction.

In order to investigate plasmoid formation in GUMICS thoroughly, a large number of simulations were run with various IMF rotation parameters:

- 8 different starting clock angles in 45 degree intervals
- clockwise and counter clockwise rotation, 180 degree flip
- rotation speed of 10 and 20 degrees per minute
- one rotation or two consecutive rotations (first clockwise then counter clockwise)

These simulations show that in GUMICS plasmoid formation is highly dependent on the details of IMF rotation. Figure 6.1 shows the structure of the closed magnetic field line region in the GSE coordinate system after a 180 degree clockwise rotation of the IMF at 20 degrees per minute with different starting clock angles. The red cells show the plasmoid identified by an automatic algorithm presented in Article IV while the blue cells show the rest of the closed magnetic field line region. As can be seen, a plasmoid forms only in half of the cases, mostly when the IMF rotates past 0 degrees i.e. starts from < 0 degrees when rotating clock wise. When a plasmoid forms its structure also depends highly on the starting angle of the rotation as only the plasmoids formed from starting angles of -45 and -90 degrees look similar.

The plasmoid structure is mirrored with respect to the dipole axis, which is in the $+Z_{GSE}$ direction in these runs, whenever the starting angle is also mirrored and the direction of IMF rotation is reversed. Figure 6.2 shows the structure of the closed magnetic field line region in the same format as Figure 6.1 with counter clockwise rotation of the IMF. The clearest example of mirroring with respect to the dipole axis can be seen when the starting angle is 135 degrees away from 0 degrees and the rotation is towards 0 degrees. These are starting angles -135 and 135 degrees in Figures 6.1 and 6.2 respectively. In a global MHD model this mirroring effect arises naturally from the symmetry of the system and the MHD equations. In the case of

e.g. the Vlasov equation, on the other hand, the symmetry is broken by the Lorentz force. Nevertheless, as the size of the system in this case is significantly larger than ion gyro radius, it would seem likely that the symmetry would also be visible in a kinetic model.

6.2.1 Corroboration from observations

There appears to be observational corroboration for the plasmoid formation hypothesis presented in Article IV. Figure 6.3 shows a series of images of a bending arc (hook-shaped arc partially attached to the auroral oval) recorded on 1999-01-19 around 05:00 UT by the Polar satellite's UVI instrument [Kullen et al., 2002] on the left and the GUMICS magnetic field topology at the inner boundary of the magnetosphere from a synthetic run with a similar IMF rotation on the right. During this event the IMF clock angle rotates counter clockwise from about 140 degrees to -40 degrees some 70 min prior to formation of the bending arc which is consistent with synthetic GUMICS results where the plasmoid forms around 60 min after the start of IMF rotation. In GUMICS the plasmoid field lines which map into the polar cap inside the auroral oval show a structure and behavior quite similar to the bending arc observed by Polar. It is possible that particles that are accelerated by reconnection at or close to the closed field lines of the plasmoid in the tail propagate along the field lines into the ionosphere resulting in the pattern detected by Polar. Even though reconnection is a purely numerical effect in ideal MHD, the structure of the magnetic field might represent reality quite well since in this case plasmoid formation is more due to the frozen in condition of the magnetic field and is initiated by the solar wind pushing the lobe magnetic field lines downstream from the Earth (Article IV). More observational examples would be needed in order to corroborate this further along with simulations of those events instead of synthetic simulations. Test particle simulations would also provide additional insight into the ionospheric signature of plasmoids.

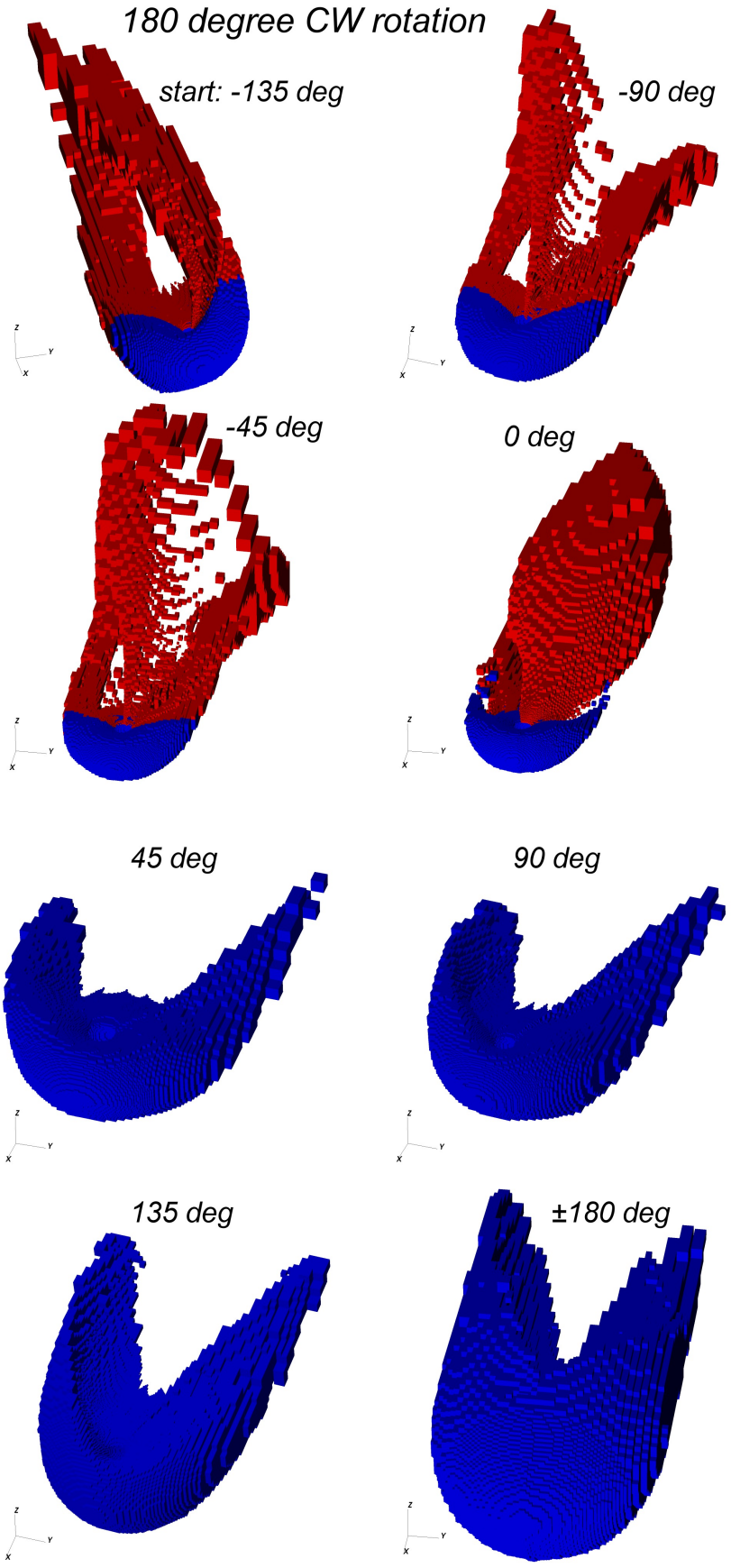

Figure 6.1: Structure of the closed field line region in GUMICS with different starting clock angles of a 180 degree clockwise rotation of the IMF. Automatically identified plasmoid cells are colored in red. The width of the blue region, i.e. its length in direction perpendicular to the Sun-Earth line, is about 30 R_E in each image.

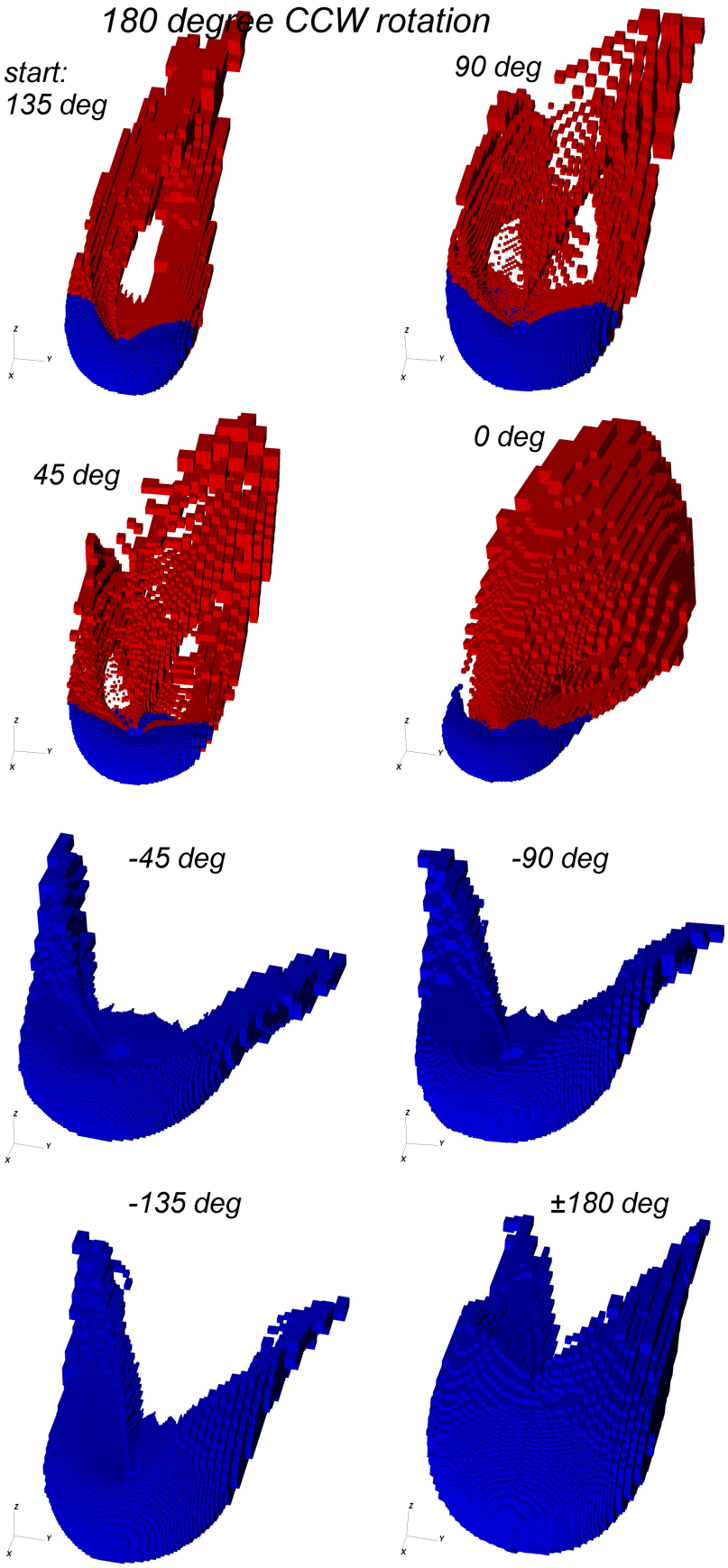

Figure 6.2: Same as Figure 6.1 but for counter clockwise rotation of IMF.

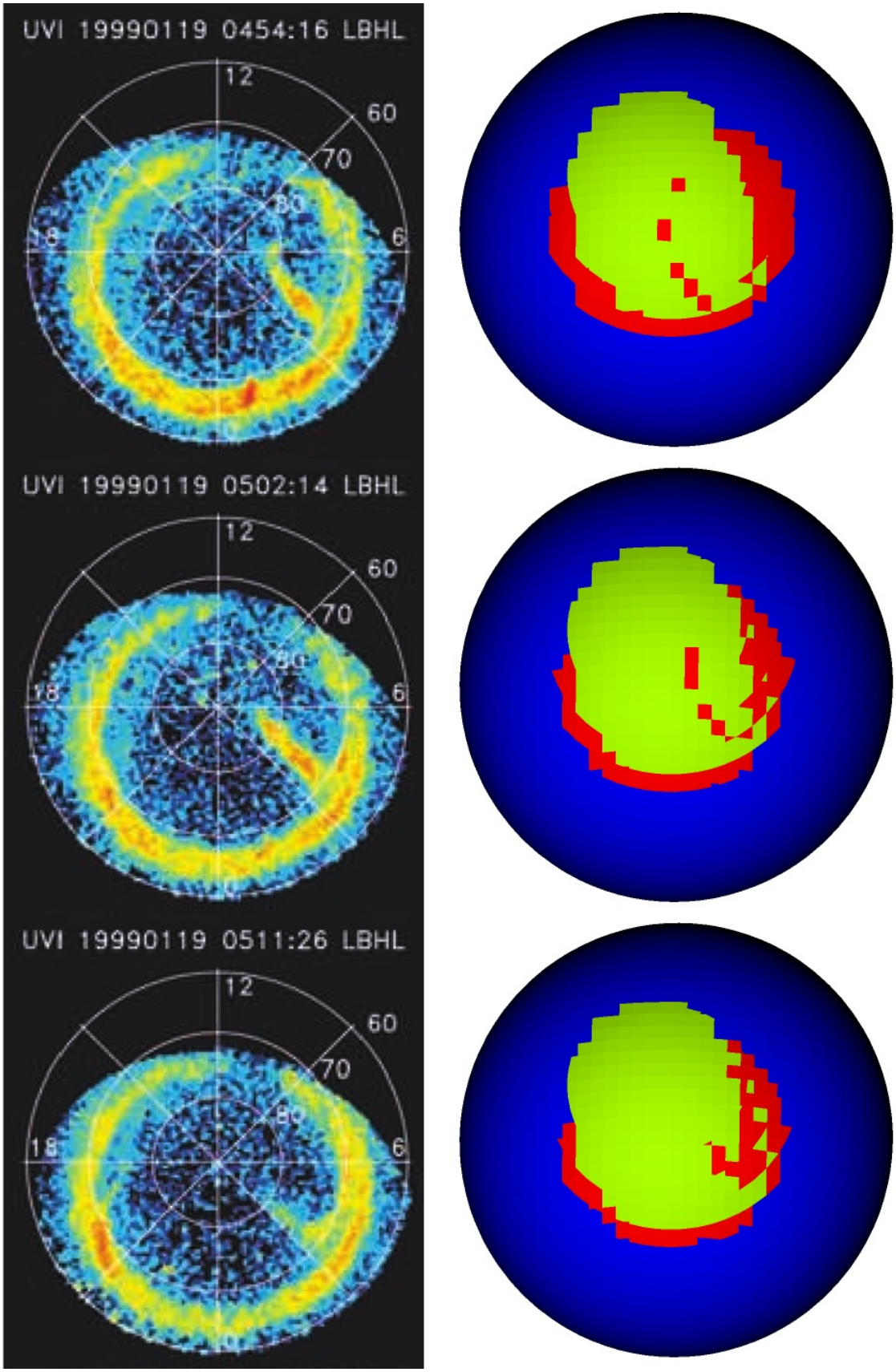

Figure 6.3: Left: UVI instrument observation of the Northern hemisphere on 1999-01-19 at around 05:00 UT from Kullen et al. [2002]. Right: Magnetic field topology at the inner boundary of the magnetosphere in GUMICS, closed field lines are shown in blue and red, lobe field lines in green. Automatically identified plasmoid field lines are shown in red.

Chapter 7

Conclusions

> I didn't come here to tell you how this is going to end. I came here to tell you how it's going to begin.
>
> Neo in The Matrix

In Article I the parallel grid library dccrg was introduced which allows rapid and flexible development of simulations based on various methods (e.g. FDM, FEM and FVM) utilizing a logically cartesian grid. Support for several novel features and excellent scalability up to 32 k processes in MHD tests with a static grid were demonstrated. The novel features of dccrg are:

- Arbitrary data stored in grid cells, such as a variable number of particles in a PIC model
- Transparent updates of cell data between processes for neighboring cells, implemented efficiently with MPI Data types
- Ease of use, a complete parallel program playing Conway's Game of Life can be implemented using dccrg with less than 60 lines of code

Additionally, the usability of run-time AMR was demonstrated in an MHD blast wave test which required an order of magnitude fewer cells than without AMR for the same effective resolution. Currently dccrg is used in the global magnetospheric hybrid Vlasov model Vlasiator and the parallel version of the global magnetospheric MHD model GUMICS. Dccrg was released as free software licensed under the GNU LGPLv3 and is available at https://gitorious.org/dccrg.

In Article II the first global 6d magnetospheric test of the hybrid Vlasov model Vlasiator was carried out in order to assess the quality of the solver for the Vlasov equation. In this test the electric and magnetic fields accelerating the plasma were not solved self-consistently but were taken from the results of the global MHD model GUMICS. The test indicated that the solver works correctly as the overall solution follows that of GUMICS, as expected. For example the structure of the bow shock is reproduced by Vlasiator and the density at the sunward edge of the bow shock is reasonable when taking into account the fact that Vlasiator used four times lower

resolution in ordinary space compared to GUMICS (1 and 0.25 R_E cells respectively). Kinetic effects related to reconnection, only possible when modeling the plasma using e.g. the Vlasov equation, were also possibly seen although other explanations for the differences between Vlasiator and GUMICS such as ionospheric outflow present in GUMICS could not be ruled out completely. The test also showed that a global hybrid Vlasov simulation of the Earth's entire magnetosphere is feasible. Since then many additional optimizations have been implemented which allow 5d simulations of the magnetosphere with spatial scales comparable to ion inertial length [von Alfthan et al., 2013b].

In Article III, for the first time to my knowledge, a validation of four global magnetospheric MHD models was carried out by simulating a single event with multiple magnetospheric substorms. Results of the BATS-R-US, GUMICS, LFM and OpenGGCM models were compared against the magnetic field measurements of Cluster, Geotail and Wind spacecraft and the cross polar cap potential derived from Super Dual Auroral Radar Network measurements. In order to estimate model performance objectively and quantitatively, the results were compared against measurements using the correlation coefficient and prediction efficiency metrics. It was shown that, overall, model performance is good on the dayside and decreases steadily downstream from the Earth. In the far tail model predictions are worse than using an average value of the magnetic field for the prediction with the exception of B_z which could be due to the Earth's dipole pointing in that direction on average. Additionally it was found that increasing model resolution or coupling of an additional physics module does not necessarily increase model performance in the magnetosphere.

Article IV presented a new hypothesis for the formation of plasmoids in the Earth's magnetotail. Based on GUMICS simulation results of an event with multiple substorms it was proposed that large plasmoids form in the magnetotail when the interplanetary magnetic field (IMF) rotates in a certain way in the plane perpendicular to the Sun-Earth line. The plasmoids are hypothesized to form due to a combination of the large scale structure of the magnetic field formed by the IMF rotation being advected downstream past the Earth and the fact that part of the magnetic field is at the same time frozen into the plasma behind the Earth. In Article III the plasmoid formation hypothesis was further supported by the simulation results of the same event from the BATS-R-US and, to a lesser extent, LFM models, while the OpenGGCM model did not support the hypothesis. Here plasmoid formation and structure in GUMICS is investigated statistically by varying the IMF rotation parameters in synthetic runs and it is found that indeed IMF rotation parameters have a significant influence on the formation of plasmoids and their resulting structure in the magnetotail. Furthermore, corroborating evidence is presented for one specific case from observations of the UV imager instrument aboard the Polar spacecraft.

This thesis describes the process of developing numerical space weather models and using such models to study the near-Earth space. A complete model development chain is presented starting from initial planning and design, to distributed memory parallelization and optimization, and finally testing, verification and validation of models. A hypothesis for a new mechanism of plasmoid formation in the

Earth's magnetotail is formulated based on the results of the global magnetospheric MHD model GUMICS.

7.1 Future prospects

> Who would have thought even five years ago (1991) that a world-class operating system could coalesce as if by magic out of part-time hacking by several thousand developers scattered all over the planet, connected only by the tenuous strands of the Internet?
>
> E.S. Raymond in The Cathedral and the Bazaar

There is a great need for kinetic modeling of the Earth's entire magnetosphere but the computational cost of existing models can be prohibitive. Numerical models capable of run-time adaptive physics have been shown to work for both high Mach number neutral flow and laboratory plasmas and to reduce the computational cost of modeling dramatically. The experience gained while preparing this thesis includes all essential skills for developing a space weather model with run-time adaptive physics that is able to resolve kinetic effects of the plasma while being computationally much less demanding than current global kinetic models of Earth's magnetosphere. The latest C++ standard approved in 2011 includes several features highly relevant for numerical model development. For example, with support for concurrency, one can take advantage of multi core CPUs efficiently while maintaining readability and maintainability. Also, variadic templates allow one to combine several numerical models without modifying any existing code while also maintaining performance. On the other hand the large number of available free [1] MHD software, for example, along with numerous sites providing public version control, issue tracking and other tools for collaboration can make international cooperation in numerical model development easier than ever before. By using proper software design and development techniques a sophisticated numerical model could be developed in an exceptionally short time. Finally, developing a numerical model as free and open source software allows anyone to examine the hypothesis represented by the software [Oreskes et al., 1994] as required by the scientific method.

[1] Free as in speech not beer (https://www.gnu.org/philosophy/free-sw.html)

Internet bibliography

Mark Buxton. Haswell New Instruction Descriptions Now Available!, 2011. URL http://software.intel.com/en-us/blogs/2011/06/13/haswell-new-instruction-descriptions-now-available. retrieved on 2013-04-22.

Eric R. Christian. The Earth's magnetosphere, 2012. URL http://helios.gsfc.nasa.gov/magneto.jpg. retrieved on 2013-05-25.

Tim Daly. axiom, 2010. URL http://www.axiom-developer.org. retrieved on 2013-05-31.

I. Honkonen, A. Sandroos, and M. Palmroth. Towards a global hybrid Vlasov magnetospheric model: a test particle simulation, 2011. URL http://presentations.copernicus.org/EGU2011-6296_presentation.pdf. Poster presentation in EGU general assembly 2011.

Ilja Honkonen. poisson_solve.hpp, 2013. URL https://gitorious.org/dccrg/dccrg/blobs/4347d45159ad99c050674e491a6a78822d4e790a/tests/poisson/poisson_solve.hpp#line209. retrieved on 2013-06-02.

IAU Commission. RESOLUTION B2, 2012. URL http://www.iau.org/static/resolutions/IAU2012_English.pdf. retrieved on 2013-05-25.

Intel. 8087 Math CoProcessor, 1989. URL http://www.datasheetcatalog.org/datasheets/2300/45014_DS.pdf. retrieved on 2013-05-03.

Andreas Kloeckner. Python wrapper for OpenCL, 2012. URL https://pypi.python.org/pypi/pyopencl. retrieved on 2013-06-02.

David Levinthal. Performance Analysis Guide for Intel Core i7 Processor and Intel Xeon 5500 processors, 2009. URL http://software.intel.com/sites/products/collateral/hpc/vtune/performance_analysis_guide.pdf. retrieved on 2013-05-31.

Maplesoft. Maple 17, 2013. URL http://www.maplesoft.com/products/maple. retrieved on 2013-05-31.

Frédéric Michel, Dennis Gallagher, and Leslie Mullen. Artist's concept of the Earth's magnetosphere, 2010. URL https://commons.wikimedia.org/wiki/File:Magnetosphere_Levels.svg. retrieved on 2013-05-25.

David A. Randall. The Shallow Water Equations, 2006. URL http://kiwi.atmos.colostate.edu/group/dave/pdf/ShallowWater.pdf. retrieved on 2013-05-11.

The Imagine Team. Imagine the Universe! Stars, 2013. URL http://imagine.gsfc.nasa.gov/docs/science/know_l2/stars.html. retrieved on 2013-07-19.

TOP500.Org. Performance Development, 2013. URL http://top500.org/statistics/perfdevel. retrieved on 2013-02-26.

Martin H. Weik. The ENIAC Story, 1961. URL http://ftp.arl.army.mil/~mike/comphist/eniac-story.html. retrieved on 2013-02-21.

Eric W. Weisstein. Parabola, 2013. URL http://mathworld.wolfram.com/Parabola.html. retrieved on 2013-05-27.

Bibliography

Aad et al. The ATLAS Simulation Infrastructure. *The European Physical Journal C*, 70:823–874, 2010. ISSN 1434-6044. doi: 10.1140/epjc/s10052-010-1429-9. URL http://dx.doi.org/10.1140/epjc/s10052-010-1429-9.

Tom Abel, Michael L. Norman, and Piero Madau. Photon-conserving Radiative Transfer around Point Sources in Multidimensional Numerical Cosmology. *The Astrophysical Journal*, 523(1):66, 1999. URL http://stacks.iop.org/0004-637X/523/i=1/a=66.

W. Richards Adrion, Martha A. Branstad, and John C. Cherniavsky. Validation, Verification, and Testing of Computer Software. *ACM Comput. Surv.*, 14(2):159–192, June 1982. ISSN 0360-0300. doi: 10.1145/356876.356879. URL https://www.cs.drexel.edu/~jhk39/teaching/cs576su06/week1Readings/adrion.pdf.

Inc. Advanced Micro Devices. AMD Athlon MP Processor Model 6 Data Sheet, 2001. URL http://support.amd.com/us/Processor_TechDocs/24685.pdf. retrieved on 2013-05-19 from Google cache.

A.T. Aikio, T. Pitkänen, I. Honkonen, M. Palmroth, and O. Amm. IMF effect on the polar cap contraction and expansion during a period of substorms. *Annales Geophysicae*, 31(6):1021–1034, 2013. doi: 10.5194/angeo-31-1021-2013. URL http://www.ann-geophys.net/31/1021/2013/.

H. Alfvén. Existence of Electromagnetic-Hydrodynamic Waves. *Nature*, 150(3805):405–406, 1942. doi: 10.1038/150405d0.

Håkan Andréasson. The Einstein-Vlasov system/kinetic theory. *Living Reviews in Relativity*, 14(4), 2011. doi: 10.12942/lrr-2011-4. URL http://www.livingreviews.org/lrr-2011-4.

Chandrasekhar R. Anekallu, M. Palmroth, Hannu E.J. Koskinen, E. Lucek, and I. Dandouras. Spatial variation of energy conversion at the Earth's magnetopause: Statistics from Cluster observations. *Journal of Geophysical Research: Space Physics*, pages 1–13, 2013. ISSN 2169-9402. doi: 10.1002/jgra.50233. URL http://dx.doi.org/10.1002/jgra.50233.

M. Ashour-Abdalla, M. El-Alaoui, M.L. Goldstein, M. Zhou, D. Schriver, R. Richard, R. Walker, M.G. Kivelson, and K.-J. Hwang. Observations and simu-

lations of non-local acceleration of electrons in magnetotail magnetic reconnection events. *Nature Physics*, 7:360–365, April 2011. doi: 10.1038/nphys1903.

Ivo Babuska and J.Tinsley Oden. Verification and validation in computational engineering and science: basic concepts. *Computer Methods in Applied Mechanics and Engineering*, 193(36-38):4057 – 4066, 2004. ISSN 0045-7825. doi: 10.1016/j.cma.2004.03.002. URL `http://www.sciencedirect.com/science/article/pii/S0045782504001781`.

Per Bak, Kan Chen, and Michael Creutz. Self-organized criticality in the 'Game of Life'. *Nature*, 342(6251):780–782, 1989. doi: 10.1038/342780a0.

D.N. Baker, S.G. Kanekal, V.C. Hoxie, M.G. Henderson, X. Li, H.E. Spence, S.R. Elkington, R.H.W. Friedel, J. Goldstein, M.K. Hudson, G.D. Reeves, R.M. Thorne, C.A. Kletzing, and S.G. Claudepierre. A Long-Lived Relativistic Electron Storage Ring Embedded in Earth's Outer Van Allen Belt. *Science*, 340(6129): 186–190, 2013. doi: 10.1126/science.1233518. URL `http://www.sciencemag.org/content/340/6129/186.abstract`.

Christopher Balch, Doug Biesecker, Larry Combs, Misty Crown, Kent Doggett, Joseph Kunches, Howard Singer, and David Zezula. Halloween space weather storms of 2003. Technical memorandum oar sec-88, NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION, 2004. URL `http://www.swpc.noaa.gov/Services/HalloweenStorms_assessment.pdf`.

W. Bangerth, R. Hartmann, and G. Kanschat. deal.II - A general-purpose object-oriented finite element library. *ACM Trans. Math. Softw.*, 33(4), August 2007. ISSN 0098-3500. doi: 10.1145/1268776.1268779. URL `http://doi.acm.org/10.1145/1268776.1268779`.

G.K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge Mathematical Library. Cambridge University Press, 2000. ISBN 9780521663960.

Thomas W. Baumgarte and Stuart L. Shapiro. General relativistic magnetohydrodynamics for the numerical construction of dynamical spacetimes. *The Astrophysical Journal*, 585(2):921, 2003. URL `http://stacks.iop.org/0004-637X/585/i=2/a=921`.

Kent Beck. *Test Driven Development*. Addison-Wesley Professional, 2002. ISBN 978-0321146533.

J. Birn, J.F. Drake, M.A. Shay, B.N. Rogers, R.E. Denton, M. Hesse, M. Kuznetsova, Z.W. Ma, A. Bhattacharjee, A. Otto, and P.L. Pritchett. Geospace Environmental Modeling (GEM) Magnetic Reconnection Challenge. *Journal of Geophysical Research: Space Physics*, 106(A3):3715–3719, 2001. ISSN 2156-2202. doi: 10.1029/1999JA900449. URL `http://dx.doi.org/10.1029/1999JA900449`.

X. Blanco-Cano, N. Omidi, and C.T. Russell. Global hybrid simulations: Foreshock waves and cavitons under radial interplanetary magnetic field geometry. *Journal*

of Geophysical Research: Space Physics, 114(A1), 2009. ISSN 2156-2202. doi: 10.1029/2008JA013406. URL `http://dx.doi.org/10.1029/2008JA013406`.

Léonard Bolduc. GIC observations and studies in the Hydro-Québec power system. *Journal of Atmospheric and Solar-Terrestrial Physics*, 64(16):1793 – 1802, 2002. ISSN 1364-6826. doi: http://dx.doi.org/10.1016/S1364-6826(02)00128-1. URL `http://www.sciencedirect.com/science/article/pii/S1364682602001281`.

J.U. Brackbill and D.C. Barnes. The effect of nonzero product of magnetic gradient and B on the numerical solution of the magnetohydrodynamic equations. *Journal of Computational Physics*, 35:426–430, May 1980. doi: 10.1016/0021-9991(80)90079-0.

R. Brightwell, K. Pedretti, and K.D. Underwood. Initial performance evaluation of the Cray SeaStar interconnect. In *High Performance Interconnects, 2005. Proceedings. 13th Symposium on*, pages 51–57, 2005. doi: 10.1109/CONECT.2005.24. URL `http://www.sandia.gov/~ktpedre/copyrighted-papers/01544577.pdf`.

Stephen W. Bruenn, Anthony Mezzacappa, W. Raphael Hix, Eric J. Lentz, O.E. Bronson Messer, Eric J. Lingerfelt, John M. Blondin, Eirik Endeve, Pedro Marronetti, and Konstantin N. Yakunin. Axisymmetric ab initio core-collapse supernova simulations of 12-25 $m_{\odot}$ stars. *The Astrophysical Journal Letters*, 767(1):L6, 2013. URL `http://stacks.iop.org/2041-8205/767/i=1/a=L6`.

L.F. Burlaga, N.F. Ness, and M.H. Acuña. Magnetic Fields in the Heliosheath: Voyager 1 Observations. *The Astrophysical Journal*, 642(1):584, 2006. URL `http://stacks.iop.org/0004-637X/642/i=1/a=584`.

Carsten Burstedde, Lucas C. Wilcox, and Omar Ghattas. `p4est`: Scalable Algorithms for Parallel Adaptive Mesh Refinement on Forests of Octrees. *SIAM Journal on Scientific Computing*, 33(3):1103–1133, 2011. doi: 10.1137/100791634.

S. Chandrasekhar. *Radiative Transfer*. Dover Publications, 1960. ISBN 0-486-60590-6. URL `http://ia701200.us.archive.org/30/items/RadiativeTransfer/Chandrasekhar-RadiativeTransfer.pdf`.

G. Chisham, M. Lester, S.E. Milan, M.P. Freeman, W.A. Bristow, A. Grocott, K.A. McWilliams, J.M. Ruohoniemi, T.K. Yeoman, P.L. Dyson, R.A. Greenwald, T. Kikuchi, M. Pinnock, J.P.S. Rash, N. Sato, G.J. Sofko, J.-P. Villain, and A.D.M. Walker. A decade of the Super Dual Auroral Radar Network (SuperDARN): scientific achievements, new techniques and future directions. *Surveys in Geophysics*, 28(1):33–109, 2007. ISSN 0169-3298. doi: 10.1007/s10712-007-9017-8. URL `http://dx.doi.org/10.1007/s10712-007-9017-8`.

Demetrios Christodoulou. The Euler Equations of Compressible Fluid Flow. *Bulletin of the American Mathematical Society*, 44(4):581–602, 2007. doi: 10.1090/S0273-0979-07-01181-0. URL `http://www.ams.org/journals/bull/2007-44-04/S0273-0979-07-01181-0/home.html`.

P.J. Coleman, Leverett Davis, and C.P. Sonett. Steady Component of the Interplanetary Magnetic Field: Pioneer V. *Phys. Rev. Lett.*, 5:43–46, Jul 1960. doi: 10.1103/PhysRevLett.5.43. URL http://link.aps.org/doi/10.1103/PhysRevLett.5.43.

Richard Courant, K. Friedrichs, and H. Lewy. On the partial difference equations of mathematical physics. *IBM J. Res. Dev.*, 11:215–234, 1967. doi: 10.1147/rd.112.0215. URL http://www.stanford.edu/class/cme324/classics/courant-friedrichs-lewy.pdf.

Andrew J. Cunningham, Adam Frank, Peggy Varnière, Sorin Mitran, and Thomas W. Jones. Simulating Magnetohydrodynamical Flow with Constrained Transport and Adaptive Mesh Refinement: Algorithms and Tests of the AstroBEAR Code. *The Astrophysical Journal Supplement Series*, 182(2):519, 2009. URL http://stacks.iop.org/0067-0049/182/i=2/a=519.

W. Daughton, J. Scudder, and H. Karimabadi. Fully kinetic simulations of undriven magnetic reconnection with open boundary conditions. *Physics of Plasmas*, 13 (7):072101, July 2006. doi: 10.1063/1.2218817.

L.P. David, A. Slyz, C. Jones, W. Forman, S.D. Vrtilek, and K.A. Arnaud. A catalog of intracluster gas temperatures. *Astrophysical Journal*, 412:479–488, August 1993. doi: 10.1086/172936.

Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. In *Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation - Volume 6*, OSDI'04, pages 10–10, Berkeley, CA, USA, 2004. USENIX Association. URL http://research.google.com/archive/mapreduce-osdi04.pdf.

J.F. Drake. Magnetic Reconnection: A Kinetic Treatment. *Washington DC American Geophysical Union Geophysical Monograph Series*, 90:155, 1995.

Marco Drewes, Sebastián Mendizabal, and Christoph Weniger. The Boltzmann equation from quantum field theory. *Physics Letters B*, 718(3):1119 – 1124, 2013. ISSN 0370-2693. doi: http://dx.doi.org/10.1016/j.physletb.2012.11.046. URL http://www.sciencedirect.com/science/article/pii/S0370269312012129.

Robert C. Duncan and Christopher Thompson. Formation of very strongly magnetized neutron stars - implications for gamma-ray bursts. *The Astrophysical Journal Letters*, 392(1):L9–L13, 1992. doi: 10.1086/186413. URL http://adsabs.harvard.edu/cgi-bin/nph-data_query?bibcode=1992ApJ...392L...9D&link_type=ARTICLE&db_key=AST.

J.W. Dungey. Interplanetary magnetic field and the auroral zones. *Phys. Rev. Lett.*, 6:47–48, Jan 1961. doi: 10.1103/PhysRevLett.6.47. URL http://link.aps.org/doi/10.1103/PhysRevLett.6.47.

J.W. Dungey. The length of the magnetospheric tail. *Journal of Geophysical Research*, 70(7):1753–1753, 1965. ISSN 2156-2202. doi: 10.1029/JZ070i007p01753. URL `http://dx.doi.org/10.1029/JZ070i007p01753`.

E. Engwall, A.I. Eriksson, C.M. Cully, M. André, R. Torbert, and H. Vaith. Earth's ionospheric outflow dominated by hidden cold plasma. *Nature Geoscience*, 2(1): 24–27, 2009. ISSN 1752-0894. doi: 10.1038/ngeo387. URL `http://dx.doi.org/10.1038/ngeo387`.

J. Enkovaara, C. Rostgaard, J.J. Mortensen, J. Chen, M. Dułak, L. Ferrighi, J. Gavnholt, C. Glinsvad, V. Haikola, H.A. Hansen, H.H. Kristoffersen, M Kuisma, A.H. Larsen, L. Lehtovaara, M. Ljungberg, O. Lopez-Acevedo, P.G. Moses, J. Ojanen, T. Olsen, V. Petzold, N.A. Romero, J. Stausholm-Møller, M. Strange, G.A. Tritsaris, M. Vanin, M. Walter, B. Hammer, H. Häkkinen, G.K.H. Madsen, R.M. Nieminen, J.K. Nørskov, M. Puska, T.T. Rantala, J. Schiøtz, K.S. Thygesen, and K.W. Jacobsen. Electronic structure calculations with GPAW: a real-space implementation of the projector augmented-wave method. *Journal of Physics: Condensed Matter*, 22(25):253202, 2010. URL `http://stacks.iop.org/0953-8984/22/i=25/a=253202`.

Robert Eymard, Thierry Gallouët, and Raphaèle Herbin. *Finite Volume Methods*, volume 7 of *Handbook for Numerical Analysis*. North Holland, 2000. URL `http://www.cmi.univ-mrs.fr/%7Eherbin/PUBLI/bookevol.pdf`.

Greg Faanes, Abdulla Bataineh, Duncan Roweth, Tom Court, Edwin Froese, Bob Alverson, Tim Johnson, Joe Kopnick, Mike Higgins, and James Reinhard. Cray cascade: a scalable HPC system based on a Dragonfly network. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC '12, pages 103:1–103:9, Los Alamitos, CA, USA, 2012. IEEE Computer Society Press. ISBN 978-1-4673-0804-5. URL `http://dl.acm.org/citation.cfm?id=2388996.2389136`.

D.H. Fairfield and J. Jones. Variability of the tail lobe field strength. *Journal of Geophysical Research: Space Physics*, 101(A4):7785–7791, 1996. ISSN 2156-2202. doi: 10.1029/95JA03713. URL `http://dx.doi.org/10.1029/95JA03713`.

Katia M. Ferrière. The interstellar environment of our galaxy. *Rev. Mod. Phys.*, 73: 1031–1066, Dec 2001. doi: 10.1103/RevModPhys.73.1031. URL `http://link.aps.org/doi/10.1103/RevModPhys.73.1031`.

Adrian Flitney and Derek Abbott. A Semi-quantum Version of the Game of Life. In AndrzejS. Nowak and Krzysztof Szajowski, editors, *Advances in Dynamic Games*, volume 7 of *Annals of the International Society of Dynamic Games*, pages 667–679. Birkhäuser Boston, 2005. ISBN 978-0-8176-4362-1. doi: 10.1007/0-8176-4429-6_35. URL `http://dx.doi.org/10.1007/0-8176-4429-6_35`.

Agner Fog. C++ vector class library. Documentation for version 1.03 β, 2013. URL `http://www.agner.org/optimize/vectorclass.pdf`.

R.A. Fonseca, L.O. Silva, F.S. Tsung, V.K. Decyk, W. Lu, C. Ren, W.B. Mori, S. Deng, S. Lee, T. Katsouleas, and J.C. Adam. OSIRIS: A Three-Dimensional, Fully Relativistic Particle in Cell Code for Modeling Plasma Based Accelerators. In P.M.A. Sloot, A.G. Hoekstra, C.J.K. Tan, and J.J. Dongarra, editors, *Computational Science - ICCS 2002*, volume 2331 of *Lecture Notes in Computer Science*, pages 342–351. Springer Berlin Heidelberg, 2002. ISBN 978-3-540-43594-5. doi: 10.1007/3-540-47789-6_36. URL http://dx.doi.org/10.1007/3-540-47789-6_36.

José A Font, Pablo Cerdá-Durán, Michael Gabler, Ewald Müller, and Nikolaos Stergioulas. Relativistic MHD simulations of stellar core collapse and magnetars. *Journal of Physics: Conference Series*, 283(1):012011, 2011. URL http://stacks.iop.org/1742-6596/283/i=1/a=012011.

Matteo Frigo, Charles E. Leiserson, Harald Prokop, and Sridhar Ramachandran. Cache-oblivious algorithms. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, FOCS '99, pages 285–, Washington, DC, USA, 1999. IEEE Computer Society. ISBN 0-7695-0409-4. URL http://dl.acm.org/citation.cfm?id=795665.796479.

M. Gardner. Mathematical Games: The Fantastic Combinations of John Conway's New Solitaire Game 'Life'. *Scientific American*, 223 (4):120–123, October 1970. ISSN 0036-8733 (print), 1946-7087 (electronic). URL http://ddi.cs.uni-potsdam.de/HyFISCH/Produzieren/lis_projekt/proj_gamelife/ConwayScientificAmerican.htm.

Tamas I. Gombosi, Gábor Tóth, Darren L. De Zeeuw, Kenneth C. Hansen, Konstantin Kabin, and Kenneth G. Powell. Semirelativistic Magnetohydrodynamics and Physics-Based Convergence Acceleration. *Journal of Computational Physics*, 177(1):176 – 205, 2002. ISSN 0021-9991. doi: http://dx.doi.org/10.1006/jcph.2002.7009.

Anthony P. Goodson, Karl-Heinz Böhm, and Robert M. Winglee. Jets from Accreting Magnetic Young Stellar Objects. I. Comparison of Observations and High-Resolution Simulation Results. *The Astrophysical Journal*, 524(1):142, 1999. URL http://stacks.iop.org/0004-637X/524/i=1/a=142.

E. Gordeev, G. Facskó, V. Sergeev, I. Honkonen, M. Palmroth, P. Janhunen, and S. Milan. Verification of the GUMICS-4 global MHD code using empirical relationships. *Journal of Geophysical Research: Space Physics*, 118(6):3138–3146, 2013. ISSN 2169-9402. doi: 10.1002/jgra.50359. URL http://dx.doi.org/10.1002/jgra.50359.

Francis H. Harlow and J. Eddie Welch. Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface. *Physics of Fluids*, 8 (12):2182–2189, 1965. doi: 10.1063/1.1761178. URL http://www.cs.rpi.edu/~cutler/classes/advancedgraphics/S09/papers/harlow_welch.pdf.

E.G. Harris. Radiative transfer in dispersive media. *Phys. Rev.*, 138:B479–B485, Apr 1965. doi: 10.1103/PhysRev.138.B479. URL `http://link.aps.org/doi/10.1103/PhysRev.138.B479`.

J. Harrison. Formal verification at Intel. In *Logic in Computer Science, 2003. Proceedings. 18th Annual IEEE Symposium on*, pages 45–54, 2003. doi: 10.1109/LICS.2003.1210044.

Les Hatton. The T experiments: errors in scientific software. *Computational Science Engineering, IEEE*, 4(2):27–38, 1997. ISSN 1070-9924. doi: 10.1109/99.609829.

Alexander Heinecke and Carsten Trinitis. Cache-oblivious matrix algorithms in the age of multicores and many cores. *Concurrency and Computation: Practice and Experience*, 2012. ISSN 1532-0634. doi: 10.1002/cpe.2974. URL `http://dx.doi.org/10.1002/cpe.2974`.

M. Henon. Vlasov equation. *Astronomy & Astrophysics*, 114:211, October 1982.

Berk Hess, Carsten Kutzner, David van der Spoel, and Erik Lindahl. GROMACS 4: Algorithms for Highly Efficient, Load-Balanced, and Scalable Molecular Simulation. *Journal of Chemical Theory and Computation*, 4(3):435–447, 2008. doi: 10.1021/ct700301q. URL `http://pubman.mpdl.mpg.de/pubman/item/escidoc:588952:2/component/escidoc:588951/412029.pdf`.

Anil N. Hirani. *Discrete Exterior Calculus*. Dissertation (Ph.D.). California Institute of Technology, 2003. URL `http://resolver.caltech.edu/CaltechETD:etd-05202003-095403`.

E.W. Hones, J. Birn, S.J. Bame, G. Paschmann, and C.T. Russell. On the three-dimensional magnetic structure of the plasmoid created in the magnetotail at substorm onset. *Geophysical Research Letters*, 9(3):203–206, 1982. ISSN 1944-8007. doi: 10.1029/GL009i003p00203. URL `http://dx.doi.org/10.1029/GL009i003p00203`.

Jr. Hones, E.W. Transient phenomena in the magnetotail and their relation to substorms. *Space Science Reviews*, 23(3):393–410, 1979. ISSN 0038-6308. doi: 10.1007/BF00172247. URL `http://dx.doi.org/10.1007/BF00172247`.

S.-T. Hong. Geometrical and hydrodynamic aspects of five-dimensional Schwarzschild black hole. *ArXiv e-prints*, January 2013. URL `http://adsabs.harvard.edu/abs/2013arXiv1301.6353H`.

I. Honkonen, M. Palmroth, T.I. Pulkkinen, P. Janhunen, and A. Aikio. On large plasmoid formation in a global magnetohydrodynamic simulation. *Annales Geophysicae*, 29(1):167–179, 2011. doi: 10.5194/angeo-29-167-2011. URL `http://www.ann-geophys.net/29/167/2011/`.

I. Honkonen, S. von Alfthan, A. Sandroos, P. Janhunen, and M. Palmroth. Parallel grid library for rapid and flexible simulation development. *Computer Physics Communications*, 184(4):1297 – 1309, 2013a. ISSN 0010-4655. doi: 10.1016/

j.cpc.2012.12.017. URL http://www.sciencedirect.com/science/article/pii/S0010465512004237.

I. Honkonen, L. Rastätter, A. Grocott, A. Pulkkinen, M. Palmroth, J. Raeder, A.J. Ridley, and M. Wiltberger. On the performance of global magnetohydrodynamic models in the Earth's magnetosphere. *Space Weather*, 11(5):313–326, 2013b. ISSN 1542-7390. doi: 10.1002/swe.20055. URL http://dx.doi.org/10.1002/swe.20055.

W.J. Hughes and D.G. Sibeck. On the 3-dimensional structure of plasmoids. *Geophysical Research Letters*, 14(6):636–639, 1987. ISSN 1944-8007. doi: 10.1029/GL014i006p00636. URL http://dx.doi.org/10.1029/GL014i006p00636.

Intel. Intel Xeon Phi Coprocessor Datasheet, 2012. URL http://www.intel.com/content/dam/www/public/us/en/documents/product-briefs/xeon-phi-datasheet.pdf. retrieved on 2013-04-24.

P. Janhunen. A Positive Conservative Method for Magnetohydrodynamics Based on HLL and Roe Methods. *Journal of Computational Physics*, 160(2):649 – 661, 2000. ISSN 0021-9991. doi: 10.1006/jcph.2000.6479. URL http://space.fmi.fi/~pjanhune/papers/MHDHLLE/paper.ps.gz.

P. Janhunen and A. Sandroos. Simulation study of solar wind push on a charged wire: basis of solar wind electric sail propulsion. *Annales Geophysicae*, 25(3):755–767, 2007. doi: 10.5194/angeo-25-755-2007. URL http://www.ann-geophys.net/25/755/2007/.

P. Janhunen, H.E.J. Koskinen, and T.I. Pulkkinen. A new global ionosphere-magnetosphere coupling simulation utilizing locally varying time step. In *Third International Conference on Substorms (ICS-3)*. European Space Agency Publications Division, 1996.

P. Janhunen, M. Palmroth, T. Laitinen, I. Honkonen, L. Juusola, G. Facskó, and T.I. Pulkkinen. The GUMICS-4 global MHD magnetosphere–ionosphere coupling simulation. *Journal of Atmospheric and Solar-Terrestrial Physics*, 80(0):48 – 59, 2012. ISSN 1364-6826. doi: 10.1016/j.jastp.2012.03.006. URL http://www.sciencedirect.com/science/article/pii/S1364682612000909.

J.P. Boris. A physically Motivated Solution of the Alfvén Problem. Nrl memorandum report 2167, 1970. URL http://www.dtic.mil/dtic/tr/fulltext/u2/715774.pdf.

F. Michael Kahnert. Numerical methods in electromagnetic scattering theory. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 79-80(0):775 – 824, 2003. ISSN 0022-4073. doi: 10.1016/S0022-4073(02)00321-7. URL http://www.sciencedirect.com/science/article/pii/S0022407302003217. Electromagnetic and Light Scattering by Non-Spherical Particles.

E. Kallio and P. Janhunen. Modelling the solar wind interaction with mercury by a quasi-neutral hybrid model. *Annales Geophysicae*, 21(11):2133–2145, 2003. doi: 10.5194/angeo-21-2133-2003. URL `http://www.ann-geophys.net/21/2133/2003/`.

H. Karimabadi, D. Krauss-Varban, J.D. Huba, and H.X. Vu. On magnetic reconnection regimes and associated three-dimensional asymmetries: Hybrid, Hall-less hybrid, and Hall-MHD simulations. *Journal of Geophysical Research: Space Physics*, 109(A9), 2004. ISSN 2156-2202. doi: 10.1029/2004JA010478. URL `http://dx.doi.org/10.1029/2004JA010478`.

Yann Kempf. Numerical and physical validation of Vlasiator - A new hybrid-Vlasov space plasma simulation code. Master's thesis, University of Helsinki, 2012. URL `http://hdl.handle.net/10138/37282`.

K. Killen, N. Omidi, D. Krauss-Varban, and H. Karimabadi. Linear and nonlinear properties of ULF waves driven by ring-beam distribution functions. *Journal of Geophysical Research: Space Physics*, 100(A4):5835–5852, 1995. ISSN 2156-2202. doi: 10.1029/94JA02899. URL `http://dx.doi.org/10.1029/94JA02899`.

Benjamin Kirk, John Peterson, Roy Stogner, and Graham Carey. libMesh: a C++ library for parallel adaptive mesh refinement/coarsening simulations. *Engineering with Computers*, 22(3-4):237–254, 2006. ISSN 0177-0667. doi: 10.1007/s00366-006-0049-3. URL `http://dx.doi.org/10.1007/s00366-006-0049-3`.

V.I. Kolobov and R.R. Arslanbekov. Towards adaptive kinetic-fluid simulations of weakly ionized plasmas. *Journal of Computational Physics*, 231(3):839 – 869, 2012. ISSN 0021-9991. doi: 10.1016/j.jcp.2011.05.036. URL `http://www.sciencedirect.com/science/article/pii/S0021999111003482`.

Hannu E.J. Koskinen. *Physics of Space Storms*. Springer Praxis Books. Springer Berlin Heidelberg, 2011. doi: 10.1007/978-3-642-00319-6.

Kei Kotake, Naofumi Ohnishi, Shoichi Yamada, and Katsuhiko Sato. Toward Radiation-Magnetohydrodynamic Simulations in Core-Collapse Supernovae. *Journal of Physics: Conference Series*, 31(1):95, 2006. URL `http://stacks.iop.org/1742-6596/31/i=1/a=015`.

A. Kullen, M. Brittnacher, J.A. Cumnock, and L.G. Blomberg. Solar wind dependence of the occurrence and motion of polar auroral arcs: A statistical study. *Journal of Geophysical Research: Space Physics*, 107(A11):13–1–13–23, 2002. ISSN 2156-2202. doi: 10.1029/2002JA009245. URL `http://dx.doi.org/10.1029/2002JA009245`.

Marko Laine, Antti Solonen, Heikki Haario, and Heikki Järvinen. Ensemble prediction and parameter estimation system: the method. *Quarterly Journal of the Royal Meteorological Society*, 138(663):289–297, 2012. ISSN 1477-870X. doi: 10.1002/qj.922. URL `http://dx.doi.org/10.1002/qj.922`.

S.A. Ledvina, Y.-J. Ma, and E. Kallio. Modeling and simulating flowing plasmas and related phenomena. *Space Science Reviews*, 139(1-4):143–189, 2008. ISSN 0038-6308. doi: 10.1007/s11214-008-9384-6. URL http://dx.doi.org/10.1007/s11214-008-9384-6.

L.C. Lee, Z.F. Fu, and S.-I. Akasofu. A simulation study of forced reconnection processes and magnetospheric storms and substorms. *Journal of Geophysical Research: Space Physics*, 90(A11):10896–10910, 1985. ISSN 2156-2202. doi: 10.1029/JA090iA11p10896. URL http://dx.doi.org/10.1029/JA090iA11p10896.

Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge texts in applied mathematics. Cambridge University Press, 2002. ISBN 0-521-00924-3.

R.L. Lin, X.X. Zhang, S.Q. Liu, Y.L. Wang, and J.C. Gong. A three-dimensional asymmetric magnetopause model. *Journal of Geophysical Research: Space Physics*, 115(A4), 2010. ISSN 2156-2202. doi: 10.1029/2009JA014235. URL http://dx.doi.org/10.1029/2009JA014235.

J.G. Lyon, J.A. Fedder, and C.M. Mobarry. The Lyon-Fedder-Mobarry (LFM) global MHD magnetospheric simulation code. *Journal of Atmospheric and Solar-Terrestrial Physics*, 66(15–16):1333 – 1350, 2004. ISSN 1364-6826. doi: 10.1016/j.jastp.2004.03.020. URL http://www.sciencedirect.com/science/article/pii/S1364682604001439.

Peter MacNeice, Kevin M. Olson, Clark Mobarry, Rosalinda de Fainchtein, and Charles Packer. PARAMESH: A parallel adaptive mesh refinement community toolkit. *Computer Physics Communications*, 126(3):330 – 354, 2000. ISSN 0010-4655. doi: http://dx.doi.org/10.1016/S0010-4655(99)00501-9. URL http://www.sciencedirect.com/science/article/pii/S0010465599005019.

Claudio Mattiussi. An Analysis of Finite Volume, Finite Element, and Finite Difference Methods Using Some Concepts from Algebraic Topology. *Journal of Computational Physics*, 133(2):289 – 309, 1997. ISSN 0021-9991. doi: 10.1006/jcph.1997.5656. URL http://infoscience.epfl.ch/record/63880/files/JCP1997.pdf.

Message Passing Interface Forum. *MPI: A Message-Passing Interface Standard Version 3.0*. University of Tennessee, 2012. URL http://www.mpi-forum.org/docs/mpi-3.0/mpi30-report.pdf.

A. Mignone, G. Bodo, S. Massaglia, T. Matsakos, O. Tesileanu, C. Zanni, and A. Ferrari. PLUTO: A Numerical Code for Computational Astrophysics. *The Astrophysical Journal Supplement Series*, 170:228–242, May 2007. doi: 10.1086/513316.

T.E. Moore, M.-C. Fok, M.O. Chandler, C.R. Chappell, S.P. Christon, D.C. Delcourt, J. Fedder, M. Huddleston, M. Liemohn, W.K. Peterson, and S. Slinker. Plasma sheet and (nonstorm) ring current formation from solar and polar wind sources. *Journal of Geophysical Research: Space Physics*, 110(A2), 2005. ISSN

2156-2202. doi: 10.1029/2004JA010563. URL http://dx.doi.org/10.1029/2004JA010563.

Y. Morinishi, T.S. Lund, O.V. Vasilyev, and P. Moin. Fully Conservative Higher Order Finite Difference Schemes for Incompressible Flow. *Journal of Computational Physics*, 143(1):90 – 124, 1998. ISSN 0021-9991. doi: 10.1006/jcph.1998.5962. URL http://dx.doi.org/10.1006/jcph.1998.5962.

Ken-Ichi Nishikawa, Shinji Koide, Jun ichi Sakai, Dimitris M. Christodoulou, Hélène Sol, and Robert L. Mutel. Three-dimensional Magnetohydrodynamic Simulations of Relativistic Jets Injected along a Magnetic Field. *The Astrophysical Journal Letters*, 483(1):L45, 1997. URL http://stacks.iop.org/1538-4357/483/i=1/a=L45.

William L. Oberkampf and Timothy G. Trucano. Verification and validation in computational fluid dynamics. *Progress in Aerospace Sciences*, 38(3):209 – 272, 2002. ISSN 0376-0421. doi: 10.1016/S0376-0421(02)00005-2. URL http://www.sciencedirect.com/science/article/pii/S0376042102000052.

Naomi Oreskes, Kristin Shrader-Frechette, and Kenneth Belitz. Verification, Validation, and Confirmation of Numerical Models in the Earth Sciences. *Science*, 263(5147):641–646, 1994. doi: 10.1126/science.263.5147.641. URL http://www.sciencemag.org/content/263/5147/641.abstract.

Stephen R. Palmer and John M. Felsing. *A Practical Guide to Feature-Driven Development*. Prentice Hall, 2002. ISBN 978-0130676153.

M. Palmroth, R.C. Fear, and I. Honkonen. Magnetopause energy transfer dependence on the interplanetary magnetic field and the Earth's magnetic dipole axis orientation. *Annales Geophysicae*, 30(3):515–526, 2012. doi: 10.5194/angeo-30-515-2012. URL http://www.ann-geophys.net/30/515/2012/.

M. Palmroth, I. Honkonen, A. Sandroos, Y. Kempf, S. von Alfthan, and D. Pokhotelov. Preliminary testing of global hybrid-Vlasov simulation: Magnetosheath and cusps under northward interplanetary magnetic field. *Journal of Atmospheric and Solar-Terrestrial Physics*, 99(0):41 – 46, 2013. ISSN 1364-6826. doi: 10.1016/j.jastp.2012.09.013. URL http://www.sciencedirect.com/science/article/pii/S1364682612002349.

Anthony T. Patera. A spectral element method for fluid dynamics: Laminar flow in a channel expansion. *Journal of Computational Physics*, 54(3):468 – 488, 1984. ISSN 0021-9991. doi: http://dx.doi.org/10.1016/0021-9991(84)90128-1. URL http://www.sciencedirect.com/science/article/pii/0021999184901281.

Per Undén et al. HIRLAM-5 Scientific Documentation. Documentation, 2002. URL http://www.hirlam.org/index.php?option=com_docman&task=doc_download&gid=270&Itemid=70.

Anthony L. Peratt. Advances in numerical modeling of astrophysical and space plasmas. *Astrophysics and Space Science*, 242:93–163, 1996. ISSN 0004-640X. doi: 10.1007/BF00645112. URL http://dx.doi.org/10.1007/BF00645112.

J. Pipitone and S. Easterbrook. Assessing climate model software quality: a defect density analysis of three models. *Geoscientific Model Development*, 5(4):1009–1022, 2012. doi: 10.5194/gmd-5-1009-2012. URL http://www.geosci-model-dev.net/5/1009/2012/.

Steven J. Plimpton and Karen D. Devine. MapReduce in MPI for Large-scale graph algorithms. *Parallel Computing*, 37(9):610–632, September 2011. ISSN 0167-8191. doi: 10.1016/j.parco.2011.02.004. URL http://dx.doi.org/10.1016/j.parco.2011.02.004.

V. Pohjola and E. Kallio. On the modeling of planetary plasma environments by a fully kinetic electromagnetic global model HYB-em. *Annales Geophysicae*, 28(3):743–751, 2010. doi: 10.5194/angeo-28-743-2010. URL http://www.ann-geophys.net/28/743/2010/.

Kenneth G. Powell. An Approximate Riemann Solver for Magnetohydrodynamics. In M.Yousuff Hussaini, Bram Leer, and John Rosendale, editors, *Upwind and High-Resolution Schemes*, pages 570–583. Springer Berlin Heidelberg, 1997. ISBN 978-3-642-64452-8. doi: 10.1007/978-3-642-60543-7_23. URL http://dx.doi.org/10.1007/978-3-642-60543-7_23.

Kenneth G. Powell, Philip L. Roe, Timur J. Linde, Tamas I. Gombosi, and Darren L. De Zeeuw. A solution-adaptive upwind scheme for ideal magnetohydrodynamics. *Journal of Computational Physics*, 154(2):284 – 309, 1999. ISSN 0021-9991. doi: 10.1006/jcph.1999.6299. URL http://www.sciencedirect.com/science/article/pii/S002199919996299X.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes*. Cambridge University Press, 2007. ISBN 9780521880688.

Daniel J. Price. Smoothed particle hydrodynamics and magnetohydrodynamics. *Journal of Computational Physics*, 231(3):759 – 794, 2012. ISSN 0021-9991. doi: 10.1016/j.jcp.2010.12.011. URL http://www.sciencedirect.com/science/article/pii/S0021999110006753.

Harald Prokop. Cache-oblivious algorithms. Master's thesis, Massachusetts Institute Of Technology, 1999.

A. Pulkkinen, M. Kuznetsova, A. Ridley, J. Raeder, A. Vapirev, D. Weimer, R.S. Weigel, M. Wiltberger, G. Millward, L. Rastätter, M. Hesse, H.J. Singer, and A. Chulaki. Geospace Environment Modeling 2008-2009 Challenge: Ground magnetic field perturbations. *Space Weather*, 9(2), 2011. ISSN 1542-7390. doi: 10.1029/2010SW000600. URL http://dx.doi.org/10.1029/2010SW000600.

Joachim Raeder, Douglas Larson, Wenhui Li, Emil L. Kepko, and Timothy Fuller-Rowell. OpenGGCM Simulations for the THEMIS Mission. *Space Science Reviews*, 141(1-4):535–555, 2008. ISSN 0038-6308. doi: 10.1007/s11214-008-9421-5. URL `http://dx.doi.org/10.1007/s11214-008-9421-5`.

S. Rafler. Generalization of Conway's "Game of Life" to a continuous domain - SmoothLife. *ArXiv e-prints*, November 2011. URL `http://adsabs.harvard.edu/abs/2011arXiv1111.1567R`.

L. Rastätter, M.M. Kuznetsova, A. Vapirev, A. Ridley, M. Wiltberger, A. Pulkkinen, M. Hesse, and H.J. Singer. Geospace Environment Modeling 2008-2009 Challenge: Geosynchronous magnetic field. *Space Weather*, 9(4), 2011. ISSN 1542-7390. doi: 10.1029/2010SW000617. URL `http://dx.doi.org/10.1029/2010SW000617`.

Eric S. Raymond. *The Cathedral and the Bazaar*. O'Reilly & Associates, Inc., Sebastopol, CA, USA, 1st edition, 1999. ISBN 1565927249. URL `http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar`.

K. Rupp, F. Rudolf, and J. Weinbub. ViennaCL - A High Level Linear Algebra Library for GPUs and Multi-Core CPUs. In *Intl. Workshop on GPUs and Scientific Applications*, pages 51–56, 2010.

A. Sandroos, I. Honkonen, S. von Alfthan, and M. Palmroth. Multi-GPU Simulations of Vlasov's Equation using Vlasiator. *Parallel Computing*, 39(8):306 – 318, 2013. ISSN 0167-8191. doi: 10.1016/j.parco.2013.05.001. URL `http://www.sciencedirect.com/science/article/pii/S0167819113000574`.

Alceste Scalas, Giovanni Casu, and Piero Pili. High-performance technical computing with erlang. In *Proceedings of the 7th ACM SIGPLAN workshop on ERLANG*, ERLANG '08, pages 49–60, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-065-4. doi: 10.1145/1411273.1411281. URL `http://doi.acm.org/10.1145/1411273.1411281`.

Jeremy D. Schnittman, Julian H. Krolik, and John F. Hawley. Light Curves from an MHD Simulation of a Black Hole Accretion Disk. *The Astrophysical Journal*, 651(2):1031, 2006. URL `http://stacks.iop.org/0004-637X/651/i=2/a=1031`.

H. Schwabe. Sonnen-Beobachtungen im Jahre 1843. *Astronomische Nachrichten*, 20(495), 1843. URL `http://articles.adsabs.harvard.edu/cgi-bin/nph-iarticle_query?bibcode=1844AN.....21..233S&db_key=AST&page_ind=0&data_type=GIF&type=SCREEN_VIEW&classic=YES`.

T.E. Schwartzentruber and I.D. Boyd. A hybrid particle-continuum method applied to shock waves. *Journal of Computational Physics*, 215(2):402 – 416, 2006. ISSN 0021-9991. doi: 10.1016/j.jcp.2005.10.023. URL `http://www.sciencedirect.com/science/article/pii/S0021999105004936`.

Hari K. Sen and Arnold W. Guess. Radiation effects in shock-wave structure. *Phys. Rev.*, 108:560–564, Nov 1957. doi: 10.1103/PhysRev.108.560. URL `http://link.aps.org/doi/10.1103/PhysRev.108.560`.

V.A. Sergeev. Penetration of the By component of the IMF into the magnetotail. *Geomagnetism and Aeronomy*, 27:612–615, August 1987.

R.B. Sheldon, H.E. Spence, J.D. Sullivan, T.A. Fritz, and J. Chen. The discovery of trapped energetic electrons in the outer cusp. *Geophysical Research Letters*, 25(11):1825–1828, 1998. ISSN 1944-8007. doi: 10.1029/98GL01399. URL `http://dx.doi.org/10.1029/98GL01399`.

J.-H. Shue, P. Song, C.T. Russell, J.T. Steinberg, J.K. Chao, G. Zastenker, O.L. Vaisberg, S. Kokubun, H.J. Singer, T.R. Detman, and H. Kawano. Magnetopause location under extreme solar wind conditions. *Journal of Geophysical Research: Space Physics*, 103(A8):17691–17700, 1998. ISSN 2156-2202. doi: 10.1029/98JA01103. URL `http://dx.doi.org/10.1029/98JA01103`.

M. Sofiev, P. Siljamo, I. Valkama, M. Ilvonen, and J. Kukkonen. A dispersion modelling system SILAM and its evaluation against ETEX data. *Atmospheric Environment*, 40(4):674 – 685, 2006. ISSN 1352-2310. doi: 10.1016/j.atmosenv.2005.09.069. URL `http://www.sciencedirect.com/science/article/pii/S1352231005009271`.

James M. Stone, Thomas A. Gardiner, Peter Teuben, John F. Hawley, and Jacob B. Simon. Athena: A New Code for Astrophysical MHD. *The Astrophysical Journal Supplement Series*, 178(1):137, 2008. URL `http://stacks.iop.org/0067-0049/178/i=1/a=137`.

Q.F. Stout, D.L. De Zeeuw, T.I. Gombosi, C.P.T. Groth, H.G. Marshall, and K.G. Powell. Adaptive Blocks: A High Performance Data Structure. In *Supercomputing, ACM/IEEE 1997 Conference*, pages 57–57, 1997. doi: 10.1109/SC.1997.10027.

Gilbert Strang and George Fix. *An analysis of the finite element method*. Wellesley-Cambridge Press, Wellesley, MA, USA, 1988. ISBN 096140888X.

Tooru Sugiyama and Kanya Kusano. Multi-scale plasma simulation by the interlocking of magnetohydrodynamic model and particle-in-cell kinetic model. *Journal of Computational Physics*, 227(2):1340 – 1352, 2007. ISSN 0021-9991. doi: 10.1016/j.jcp.2007.09.011. URL `http://www.sciencedirect.com/science/article/pii/S0021999107003968`.

Yanhua Sun, Gengbin Zheng, Chao Mei, Eric J. Bohm, Terry Jones, Laxmikant V. Kalé, and James C. Phillips. Optimizing Fine-grained Communication in a Biomolecular Simulation Application on Cray XK6. In *Proceedings of the 2012 ACM/IEEE conference on Supercomputing*, Salt Lake City, Utah, November 2012. URL `http://charm.cs.illinois.edu/newPapers/12-33/paper.pdf`.

R. Teyssier. Cosmological hydrodynamics with adaptive mesh refinement. A new high resolution code called RAMSES. *Astronomy & Astrophysics*, 385:337–364, April 2002. doi: 10.1051/0004-6361:20011817.

William M. Thomas, Alex Delis, and Victor R. Basili. An analysis of errors in a reuse-oriented development environment. *Journal of Systems Software*, 38:211–224, 1995.

Chris J. Thompson, Sahngyun Hahn, and Mark Oskin. Using modern graphics architectures for general-purpose computing: a framework and analysis. In *Proceedings of the 35th annual ACM/IEEE international symposium on Microarchitecture*, MICRO 35, pages 306–317, Los Alamitos, CA, USA, 2002. IEEE Computer Society Press. ISBN 0-7695-1859-1. URL `http://dl.acm.org/citation.cfm?id=774861.774894`.

Sudarshan Tiwari, Axel Klar, and Steffen Hardt. A particle-particle hybrid method for kinetic and continuum equations. *Journal of Computational Physics*, 228 (18):7109 – 7124, 2009. ISSN 0021-9991. doi: 10.1016/j.jcp.2009.06.019. URL `http://www.sciencedirect.com/science/article/pii/S0021999109003428`.

R.A. Treumann, W. Baumjohann, and W.D. Gonzalez. Collisionless reconnection: magnetic field line interaction. *Annales Geophysicae*, 30(10):1515–1528, 2012. doi: 10.5194/angeo-30-1515-2012. URL `http://www.ann-geophys.net/30/1515/2012/`.

Gábor Tóth, Darren L. De Zeeuw, Tamas I. Gombosi, and Kenneth G. Powell. A parallel explicit/implicit time stepping scheme on block-adaptive grids. *Journal of Computational Physics*, 217(2):722 – 758, 2006. ISSN 0021-9991. doi: 10.1016/j.jcp.2006.01.029. URL `http://www.sciencedirect.com/science/article/pii/S0021999106000337`.

Gábor Tóth, Bart van der Holst, Igor V. Sokolov, Darren L. De Zeeuw, Tamas I. Gombosi, Fang Fang, Ward B. Manchester, Xing Meng, Dalal Najib, Kenneth G. Powell, Quentin F. Stout, Alex Glocer, Ying-Juan Ma, and Merav Opher. Adaptive numerical algorithms in space weather modeling. *Journal of Computational Physics*, 231(3):870 – 903, 2012. ISSN 0021-9991. doi: 10.1016/j.jcp.2011.02.006. URL `http://www.sciencedirect.com/science/article/pii/S002199911100088X`.

UPC Consortium. Unified Parallel C Language Specifications document version 1.2. Tech report lbnl-59208, Lawrence Berkeley National Lab, 2005. URL `http://www.gwu.edu/~upc/publications/LBNL-59208.pdf`.

Brian Van Straalen, Phil Colella, Daniel T. Graves, and Noel Keen. Petascale block-structured AMR applications without distributed meta-data. In *Proceedings of the 17th international conference on Parallel processing - Volume Part II*, Euro-Par'11, pages 377–386, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-23396-8. URL `http://dl.acm.org/citation.cfm?id=2033408.2033452`.

S. von Alfthan, D. Pokhotelov, Y. Kempf, S. Hoilijoki, I. Honkonen, A. Sandroos, and M. Palmroth. Vlasiator: First six-dimensional global hybrid-Vlasov simulation code for modeling near-Earth space. *Journal of Atmospheric and Solar-Terrestrial Physics*, 2013b. manuscript in preparation.

Sebastian von Alfthan, Ilja Honkonen, and Minna Palmroth. Topology aware process mapping. In Pekka Manninen and Per Öster, editors, *Applied Parallel and Scientific Computing*, volume 7782 of *Lecture Notes in Computer Science*, pages 297–308. Springer, 2013a. ISBN 978-3-642-36802-8.

Peng Wang and Tom Abel. Magnetohydrodynamic Simulations of Disk Galaxy Formation: The Magnetization of the Cold and Warm Medium. *The Astrophysical Journal*, 696(1):96, 2009. URL `http://stacks.iop.org/0004-637X/696/i=1/a=96`.

X. Wang and C. Jin. Image encryption using Game of Life permutation and PWLCM chaotic system. *Optics Communications*, 285:412–417, February 2012. doi: 10.1016/j.optcom.2011.10.010.

M.S. Warren and J.K. Salmon. A parallel hashed Oct-Tree N-body algorithm. In *Proceedings of the 1993 ACM/IEEE conference on Supercomputing*, Supercomputing '93, pages 12–21, New York, NY, USA, 1993. ACM. ISBN 0-8186-4340-4. doi: 10.1145/169627.169640. URL `http://doi.acm.org/10.1145/169627.169640`.

D.T. Welling, J. Koller, and E. Camporeale. Verification of SpacePy's radial diffusion radiation belt model. *Geoscientific Model Development*, 5(2):277–287, 2012. doi: 10.5194/gmd-5-277-2012. URL `http://www.geosci-model-dev.net/5/277/2012/`.

Hettithanthrige S. Wijesinghe and Nicolas G. Hadjiconstantinou. Discussion of hybrid atomistic-continuum methods for multiscale hydrodynamics. *International Journal for Multiscale Computational Engineering*, 2:189–202, 2004.

John H. Wise, Matthew J. Turk, Michael L. Norman, and Tom Abel. The birth of a galaxy: Primordial metal enrichment and stellar populations. *The Astrophysical Journal*, 745(1):50, 2012. URL `http://stacks.iop.org/0004-637X/745/i=1/a=50`.

Andrew M. Wissink, Richard D. Hornung, Scott R. Kohn, Steve S. Smith, and Noah Elliott. Large scale parallel structured AMR calculations using the SAMRAI framework. In *Proceedings of the 2001 ACM/IEEE conference on Supercomputing (CDROM)*, Supercomputing '01, pages 6–6, New York, NY, USA, 2001. ACM. ISBN 1-58113-293-X. doi: 10.1145/582034.582040. URL `http://doi.acm.org/10.1145/582034.582040`.

Masaaki Yamada, Russell Kulsrud, and Hantao Ji. Magnetic reconnection. *Rev. Mod. Phys.*, 82:603–664, Mar 2010. doi: 10.1103/RevModPhys.82.603. URL `http://link.aps.org/doi/10.1103/RevModPhys.82.603`.

Kane S. Yee. Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media. *IEEE Trans. Antennas and Propagation*, pages 302–307, 1966.

Article I

Parallel grid library for rapid and flexible simulation development,
Computer Physics Communications, volume 184, issue 4, 2013, pages 1297-1309.

Computer Physics Communications 184 (2013) 1297–1309

Contents lists available at SciVerse ScienceDirect

Computer Physics Communications

journal homepage: www.elsevier.com/locate/cpc

Parallel grid library for rapid and flexible simulation development☆

I. Honkonen [a,b,*], S. von Alfthan [a], A. Sandroos [a], P. Janhunen [a], M. Palmroth [a]

[a] Finnish Meteorological Institute, Helsinki, Finland
[b] Department of Physics, University of Helsinki, Helsinki, Finland

ARTICLE INFO

Article history:
Received 24 May 2012
Received in revised form
5 October 2012
Accepted 14 December 2012
Available online 29 December 2012

Keywords:
Parallel grid
Adaptive mesh refinement
Free open source software

ABSTRACT

We present an easy to use and flexible grid library for developing highly scalable parallel simulations. The distributed cartesian cell-refinable grid (dccrg) supports adaptive mesh refinement and allows an arbitrary C++ class to be used as cell data. The amount of data in grid cells can vary both in space and time allowing dccrg to be used in very different types of simulations, for example in fluid and particle codes. Dccrg transfers the data between neighboring cells on different processes transparently and asynchronously allowing one to overlap computation and communication. This enables excellent scalability at least up to 32 k cores in magnetohydrodynamic tests depending on the problem and hardware. In the version of dccrg presented here part of the mesh metadata is replicated between MPI processes reducing the scalability of adaptive mesh refinement (AMR) to between 200 and 600 processes. Dccrg is free software that anyone can use, study and modify and is available at https://gitorious.org/dccrg. Users are also kindly requested to cite this work when publishing results obtained with dccrg.

Program summary

Program title: DCCRG
Catalogue identifier: AEOM_v1_0
Program summary URL: http://cpc.cs.qub.ac.uk/summaries/AEOM_v1_0.html
Program obtainable from: CPC Program Library, Queen's University, Belfast, N. Ireland
Licensing provisions: GNU Lesser General Public License version 3
No. of lines in distributed program, including test data, etc.: 54975
No. of bytes in distributed program, including test data, etc.: 974015
Distribution format: tar.gz
Programming language: C++.
Computer: PC, cluster, supercomputer.
Operating system: POSIX.
The code has been parallelized using MPI and tested with 1–32768 processes
RAM: 10 MB–10 GB per process
Classification: 4.12, 4.14, 6.5, 19.3, 19.10, 20.
External routines: MPI-2 [1], boost [2], Zoltan [3], sfc++ [4]
Nature of problem:
Grid library supporting arbitrary data in grid cells, parallel adaptive mesh refinement, transparent remote neighbor data updates and load balancing.
Solution method:
The simulation grid is represented by an adjacency list (graph) with vertices stored into a hash table and edges into contiguous arrays. Message Passing Interface standard is used for parallelization. Cell data is given as a template parameter when instantiating the grid.
Restrictions:
Logically cartesian grid.

☆ This paper and its associated computer program are available via the Computer Physics Communication homepage on ScienceDirect (http://www.sciencedirect.com/science/journal/00104655).

* Correspondence to: P.O. Box 503, 00101 Helsinki, Finland. Tel.: +35 8503803147; fax: +35 8295394603.
E-mail addresses: ilja.honkonen@fmi.fi, ilja.honkonen@helsinki.fi (I. Honkonen).

0010-4655/$ – see front matter
doi:10.1016/j.cpc.2012.12.017

Running time:
Running time depends on the hardware, problem and the solution method. Small problems can be solved in under a minute and very large problems can take weeks. The examples and tests provided with the package take less than about one minute using default options.

In the version of dccrg presented here the speed of adaptive mesh refinement is at most of the order of 10^6 total created cells per second.

References:

[1] http://www.mpi-forum.org/.
[2] http://www.boost.org/.
[3] K. Devine, E. Boman, R. Heaphy, B. Hendrickson, C. Vaughan, Zoltan data management services for parallel dynamic applications, Comput. Sci. Eng. 4 (2002) 90–97. http://dx.doi.org/10.1109/5992.988653.
[4] https://gitorious.org/sfc++.

1. Introduction

During the rising phase of the solar cycle, it is becoming more important to understand the physics of the near-Earth space. The dynamical phenomena caused by the constant flow of magnetized collisionless plasma from the Sun creates space weather that may have harmful effects on space-borne or ground-based technological systems or on humans in space. While the physics of space weather is being studied with in situ instruments (e.g. NASA's Radiation Belt Storm Probes launched in 2012-08-30[1]) and by means of remote sensing, it is also important to model the near-Earth space with numerical simulations. The simulations can be used both as a context to the one-dimensional data sets from observations, as well as a source to discover new physical mechanisms behind observed variations. Present large scale (global) simulations are based on computationally light-weight simplified descriptions of plasma, such as magnetohydrodynamics (MHD, [1–4]). On the other hand the complexity and range of spatial scales (from less than 10^1 to over 10^6 km) in space weather physics signifies the need to incorporate particle kinetic effects in the modeled equation set in order to better model, for example, magnetic reconnection, wave–particle interactions, shock acceleration of particles, ring current, radiation belt dynamics and charge exchange (see e.g. [5] for an overview). However, as one goes from MHD towards the full kinetic description of plasma (from hybrid PIC [6] and Vlasov [7] to full PIC [8,9]), the computational demands increase rapidly, indicating that the latest high performance computing techniques need to be incorporated in the design of new simulation architectures.

As the number of cores in the fastest supercomputers increases exponentially the parallel performance of simulations on distributed memory machines is becoming crucial. On the other hand, utilizing a large number of cores efficiently in parallel is challenging especially in simulations using run-time adaptive mesh refinement (AMR). This is largely a data structure and an algorithm problem albeit specific to massively parallel physical simulations running on distributed memory machines.

In computer simulations dealing with, for example, continuous matter (a fluid) the simulated domain is discretized into a set of points or finite volumes which we will refer to as cells. At any given cell the numerical solution of a differential equation describing the problem often depends only on data within a (small) part of the simulated volume. This is true for a single time step in a solver for a hyperbolic problem or a single iteration in a solver for an elliptic problem. This spatial data dependency can be implemented implicitly in the solver function(s) or explicitly as a separate grid library used by the application.

In a simple case the number of cells in the simulation stays constant and the data dependency of each cell is identical allowing cell data to be stored in an array whose size is determined at grid creation and the spatial neighbors to be represented as indices into this array. A straightforward AMR extension of this concept is to create additional nested grids in specific parts of the simulation domain with higher resolution. By solving each grid separately and interpolating the results from finer grids into coarser grids one does not have to modify the solver functions. This technique is used extensively for example by Berger (see [10] for some of the earliest work) and by [11,12]. In the rest of this work however we will concentrate on AMR implementations in which additional overlapping grids are not created but instead cells of the initial grid are refined, i.e. replaced with multiple smaller cells.

A generic unstructured grid (as provided for example by libMESH [13]) does not admit as simple a description as above and is generally described by a directed graph in which vertices represent simulation cells and directed edges represent the data dependencies between cells. Unfortunately the nomenclature of graph theory and geometry overlap to some extent and discussing both topics simultaneously can lead to confusion. Fig. 1 shows the nomenclature we use from this point forward, the standard graph theoretical terms are given in parentheses for reference. A cell is a natural unit in simulations using the finite volume method (FVM) and hereinafter we will use the term cell instead of vertex when discussing graphs. Also an edge in FVM simulations usually refers to the edges of a cube representing the physical volume of a cell, and hence we will use the term arrow to refer to a directed edge in a graph. Furthermore we note that each cell in the grid can also represent, for example, a block of cells similarly to [3], but for the purposes of this work the actual data stored in grid cells is largely irrelevant.

Since a graph can also be used to represent the cells and arrows of grids simpler than an unstructured mesh, the question arises how does a particular program implement its graph representation of the simulated system, e.g. what simplifying assumptions have been made and how is the graph represented in memory? A popular representation in (M)HD AMR simulations is to have a fixed number of arrows directed away from each source cell and to store the arrows as native pointers to the destination cells. In case a cell does not exist all arrows pointing to it are invalidated in neighboring cells. This technique has been used with different variations by [14–17], for example.

There are several possibilities for representing the cells and arrows of a graph, for example an adjacency list or an adjacency matrix [18]. In physical simulations the number of arrows in the graph is usually of the same order as the number of cells in which

[1] http://www.nasa.gov/mission_pages/rbsp/main/index.html

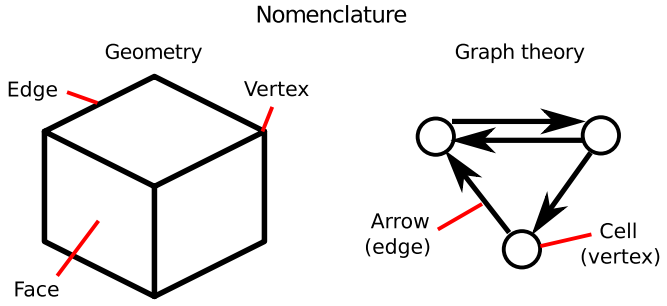

Fig. 1. The nomenclature used in this work for geometry and graph theory. Standard graph theoretical terms are given in parentheses for reference.

case a suitable representation is an adjacency list. In an adjacency list the cells of the graph are separate objects and each cell stores the arrows pointing to and/or from that cell. The cells of the graph and the arrows of each cell can be stored in different types of data structures. For example the cells are stored in a contiguous array (representing a linear octree) in [19–22], a hash table in [23] and a (doubly) linked list in [14]. On the other hand the arrows of each cell are stored in a fixed size array of native pointers in [14] and as single bits in [23].

In this work we introduce the distributed cartesian cell-refinable grid (dccrg) for rapid development of parallel simulations using, for example, finite volume or finite element methods (FEM). In dccrg the graph is represented by an adjacency list in which cells are stored into a hash table, while the arrow lists of cells are stored into contiguous arrays. We describe the details of the graph representation in Section 2. In Section 3 we describe the C++ implementation of dccrg and present its unique features with respect to other published grid codes: arbitrary data in grid cells, transparent updates of remote neighbor data, user-selectable neighborhood size for cells and ease of use. In Section 4 we test the scalability of dccrg using a variety of tests in one, two and three dimensions and draw our conclusions in Section 5.

2. Implementation of the grid graph

Dccrg represents the grid as an adjacency list in which cells are stored into a hash table. A hash table has one clear advantage over a tree when used to store the grid: cells can be accessed, inserted and deleted in constant amortized time regardless of the number of cells and their physical size and location. Thus neither the total number of cells nor the number of refinement levels affect the simulating performance of a single core. Each cell is associated with a unique id which we use as a key into the hash table. A potential drawback of a hash table is the computational cost of the hash function, but according to our tests the cost is usually not important. The time to solve one flux between two cells in the MHD tests presented in Section 4.1 is about four times larger than accessing one random cell in the hash table. The cell access time can be optimized further, for example, by storing and solving blocks of cells instead of single cells as is done in [3] and discussed further in Section 3.1.1.

2.1. Mapping cell ids to a physical location

Our cell ids are globally unique integers which offers several advantages. (1) The cell id can be calculated locally, i.e. without communication with other processes. (2) The neighbors of a cell can be stored as cell ids instead of pointers that are not consistent across computing nodes. (3) The cost of computing the hash function values is minimized. (4) The memory required for storing the cell ids is small.

Fig. 2 shows an example of mapping cells to unique ids which is done as follows: cell ids within each refinement level increase monotonically first in x coordinates, then in y and then in z, with cells at refinement level 0 represented by numbers from 1 to N_0, refinement level 1 by numbers from N_0+1 to N_0+1+N_1, etc. Cells at refinement level $l+1$ are half the size of cells at refinement level l in each dimension. Cells at equal refinement level are identical in size. Hence in three dimensions $N_0 = \frac{N_1}{8} = \frac{N_2}{64} = \cdots = n_x n_y n_z$, where n_x, n_y and n_z are the grid size in cells of refinement level 0 in the x, y and z dimensions respectively. Hereinafter cell size refers to the logical size of cells assuming a homogeneous and isotropic cartesian geometry.

When searching for the neighbors of a cell in the hash table (see Section 2.2) it is convenient to use the concept of cell indices: the location of each cell in the grid is represented by one number per dimension in the interval $[0, 2^L n_i - 1]$ where L is the maximum refinement level of the grid and n_i is n_x, n_y or n_z respectively. Fig. 2 shows the possible cell indices for an example grid with $n_x = n_y = n_z = 1$ and $L = 2$. The size of a single cell of refinement level l is 2^{L-l} indices in each dimension. A cell spanning more than one index is considered to be located at indices closest to the origin of the grid, for example cell #3 in Fig. 2 is located at indices (2, 0, 0). Similarly to [24] there is a one-to-one mapping between cell ids and cell indices plus refinement levels, e.g. in addition to its id a cell can be uniquely identified by its indices and refinement level.

In the current implementation of dccrg a cell is refined by creating all of its children; in the example grid of Fig. 2 refining cell #1 would create cells #2…#9. In principle this is not required and more complex grid structures are possible in which, for example, the grid in Fig. 2 would consist of cells #1 and #3 alone. Such an approach has been found useful by [14]. Complete refinement of cells in our case was a practical decision based on our current simulation needs and it also simplifies the neighbor searching code and enables optimizations described in the next section.

2.2. Neighbor searching

In dccrg all cells existing within a certain minimum distance from local cells (cells owned by the process) are stored in a hash table with the cell id as the key and the process owning the cell as the value. Since the mapping of cell ids to a location is unique, finding the neighbors of a cell in the hash table is straightforward: for all indices neighboring a given cell the hash table is searched for cells of all applicable refinement levels. Fig. 3 shows an example of neighbor searching in a grid with $n_x = 2, n_y = n_z = 1, L = 3$ and a neighborhood size of one. The siblings of cell #4 (#3, #7, #8, #11, #12, #15 and #16) are not shown for clarity and some potential neighbors of cell #4 in the positive x direction have also been omitted. In dccrg the cells' neighborhoods are measured in multiples of their own size, e.g. for cell #4 all cells that are (at least partially) between indices 0 and 11 inclusive in the x direction would be considered as neighbors. In order to find the neighbor(s) of cell #4 in the positive x direction the hash table is searched for the smallest cell at indices (8, 0, 0) which in this case could correspond to any of the following cells: #2, #5, #23 or #155. If cell #23 is the smallest cell found in the hash table the search can stop since it is known that the siblings of cell #23 also exist (not shown are cells #24, #31, #32, #55, #56, #63 and #64) because all children of a cell are created when a cell is refined. On the other hand if cell #155 is the smallest cell found at indices (8, 0, 0) then the search would continue at indices outside of cell #155 and its siblings, for example at indices (10, 0, 0) or (8, 2, 0).

The concept of hanging nodes or faces used in unstructured mesh codes is not directly applicable to dccrg because a single cell is the smallest unit that dccrg deals with. Since the user is responsible for defining what is stored in each cell he/she must also define, if required, the data stored at the faces, edges and vertices of cells. Hanging nodes, which are the result of cells of

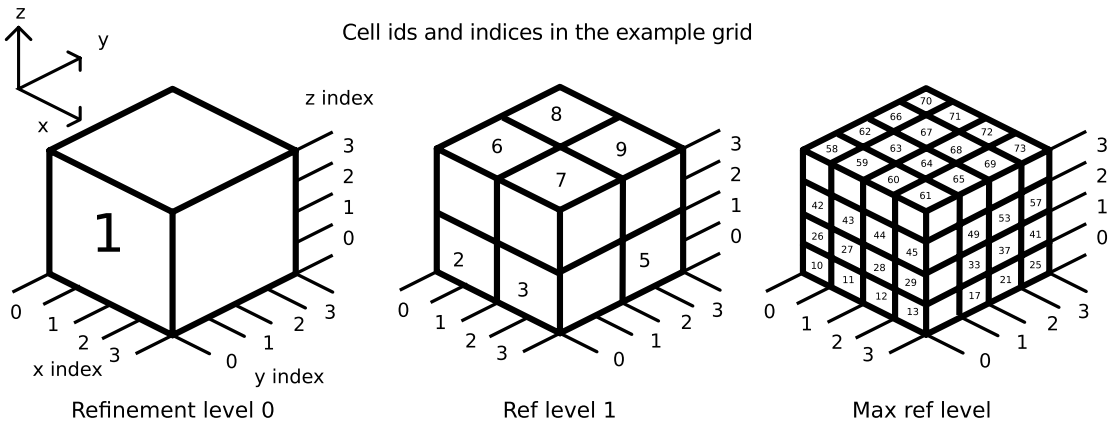

Fig. 2. An example dccrg grid of size 1 in each dimension in cells of refinement level 0, with a maximum refinement level 2, showing the ids and indices of all possible cells.

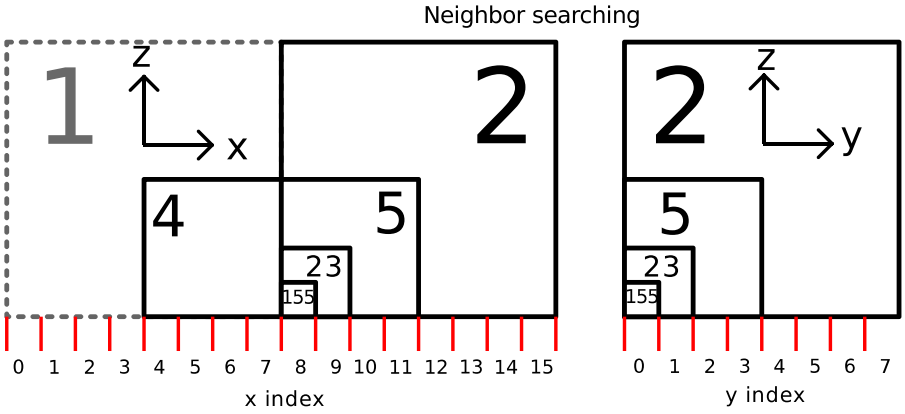

Fig. 3. An example grid illustrating neighbor searching for cell #4 in the positive *x* direction. The siblings of cell #4 are not shown for clarity and also some of its potential neighbors in positive *x* direction have been left out.

different size sharing an edge or a face, must be handled by the solvers used for a particular application. For example in GUMICS-4 [1] where the magnetic field is separated into background and perturbed components the face average background magnetic field is required when solving the flux through a face. With AMR a face average value of the background field is required for every face of every cell because, for example, if only one face average is stored per dimension in cells it would not be possible to solve the flux between cells (#4, #2) and (#1, #5) in Fig. 3. This is due to the fact that the face average value of the smaller cell must be used in both cases and it is only available if every face of every cell stores the face average field.

Currently dccrg enforces a maximum refinement level difference of 1 between neighboring cells. Hence it is sufficient to search for cells of three refinement levels $l-1 \ldots l+1$ when finding the neighbors of a cell of refinement level l. In principle the enforcement of maximum refinement level difference is not required. For example in Fig. 3 cell #4 (refinement level 1) has cell #155 (refinement level 3) as a neighbor but in the current version of dccrg such a situation is not permitted and searching the hash table for cells #2, #5 and #23 is sufficient for finding the neighbors of cell #4. This was a practical decision based on our experience with global MHD modeling of the Earth's magnetosphere using GUMICS-4. In future this restriction might be removed. A similar one is also used in [24].

Even though a maximum refinement level difference of one is enforced between neighboring cells and searching for cells in the hash table is a quick operation, in practice the ids of neighbors of local cells are also stored explicitly. As mentioned in Section 1, this neighbor information corresponds to arrows between cells in a graph and hence we will use the term arrow list to refer to the neighbor list of a cell. Dccrg stores both the arrows pointing away from and the arrows pointing to local cells. With AMR it is possible that there exists only one arrow between two cells because cells' neighborhoods are measured in units of their own size. For example in Fig. 2 with a neighborhood size of 1, cell #13 would be considered a neighbor of cell #2 but cell #2 would not be considered a neighbor of cell #13. Explicitly storing the arrows to and from local cells enables fast iteration for example by user code (solvers, reconstruction functions, etc.). In dccrg the arrow lists of local cells are stored as contiguous arrays.

2.3. Algorithmic advantages

The most important advantage that globally unique cell ids have over a traditional graph implementation using native pointers between cells is that the arrows between cells are not required to be consistent during AMR or load balancing. For example when doing AMR, a pointer-based implementation has to be careful not to leave any dangling pointers and to update the pointers of all nearby cells in the correct order so as not to lose access to any cell. On the other hand with unique cell ids the arrow lists of cells can simply be emptied when needed and new neighbors searched in the hash table as described in Section 2.2. In addition to being easy to implement this method also admits simple thread-based parallelization inside dccrg due to different threads modifying only the arrow lists of different cells.

It is also advantageous to use unique cell ids in arrow lists instead of pointers in a parallel program running on distributed memory hardware. In this environment a pointer to a neighboring

cell is only valid on one process and cannot be used to refer to the same neighbor on other processes. On the other hand the same unique cell id can be used by all processes to refer to the same neighbor regardless of its actual location in memory.

3. Implementation

A separate grid library is a natural abstraction probably for any physical simulation but especially for simulations using FVM where the concept of a grid and its cells' data dependencies are easy to define and implement. Thus following good software development practice dccrg is implemented independently of any specific physical problem or its solver, while still providing the flexibility required for various types of simulations, for example (M)HD, advection (e.g. Vlasov) and kinetic.

Dccrg is written in C++ which allows us to easily separate low level functionality of dccrg into subclasses which higher-level classes can use, thus also benefiting from a modular internal implementation, a technique also used in [13]. For example the physical geometry of the grid is handled by a separate class which is also given to dccrg as a template parameter. This allows one to easily extend the grid geometries supported by dccrg. In the default homogeneous and cartesian geometry cells of the same refinement level are identical in size in each dimension.[2]

Here we describe the unique user-visible features of dccrg with respect to other grid codes and also present important features of the serial and parallel implementation.

3.1. Unique features

3.1.1. Arbitrary data in grid cells

The most important feature distinguishing dccrg from other grid codes is the possibility of trivially storing data of arbitrary type and size in the grid's cells by simply giving the class which is stored in the cells as a template parameter to dccrg when creating an instance of the grid. This also allows a single simulation to have several independent parallel grids with different geometries and different types of data stored in the grids' cells'. The amount of data can also vary between different cells of the same grid and in the same cell as a function of time. This is required for example in kinetic simulations where not only does the total number of particles change but also the number of particles in each grid cell varies. In dccrg this is handled by each instance of the user's cell data class providing a MPI datatype corresponding to the data to be sent from or received by that particular cell. An example of this is presented in Section 3.1.4.

Completely arbitrary cell data can also be transferred between processes if the cell data class provides a serialize function which the MPI bindings of boost library will use for transferring cell data between processes.[3] Although this method of transferring data between processes is the most general it is also the slowest since data is first copied into a contiguous buffer by serialization and subsequently transferred by MPI resulting in at least one additional copy of the data being created compared to a pure MPI transfer. This is also the case in the SAMRAI framework [25] which supports transferring arbitrary patch data using the same technique.

3.1.2. Automatic remote neighbor updates

Dccrg can automatically transfer cell data between processes both for remote neighbor data updates and load balancing using simple function calls. Furthermore whenever cell data is sent between processes either one MPI message per cell can be used or, similarly to [25], all cells being sent to another process can be transferred using a single MPI message. Updating the remote neighbor data between processes is possible using several methods. The simplest one is the synchronous update function that updates the remote neighbor data between processes and returns once transfers have completed (see Section 3.1.4). The most fine-grained communication currently supported can be used by calling a separate function for initiating transfers and functions that wait for the sends and receives to complete respectively. A typical usage scenario would consist of the following:

1. Start transferring remote neighbor data.
2. Solve the inner cells of the simulation (cells without remote neighbors).
3. Wait for the data from other processes to arrive.
4. Solve the outer cells of the simulation (cells with at least one neighbor on another process).
5. Wait for the data from this process to be sent.

The MHD scalability tests we present in Section 4.1 use this procedure with the exception that step 5 is executed before step 4 due to the technical implementation of the GUMICS-4 MHD solver.

3.1.3. User-selectable neighborhood size

As mentioned in Section 1 the size of cells' neighborhood in simulations is highly problem/solver dependent. Specifically the problem/solver used in the simulation dictates the distance from which data is required at a cell in order to advance the simulation for one time step or one iteration in that cell. In many previously published grid codes the size of cells' neighborhood is restricted to 1 either explicitly or implicitly. For example in [24] in three dimensions a block (a single cell from the point of view of the grid) has 6 neighbors and it is assumed that a block consists of such a number of simulation cells that, for example, a solver needing data from a distance of 3 simulation cells can obtain that data from the neighboring block. Other examples are [21,19,15,13,16,17]. Dccrg supports an arbitrarily large neighborhood chosen by the user when the grid is initialized. The size of the neighborhood can be any unsigned integer and all other cells within that distance of a cell (in units of the size of the cell itself) will be considered as neighbors of the cell. This enables the use of high-order solvers with the added possibility of refining each neighboring cell individually. Naturally one can also store a sufficiently large block of simulation cells in one dccrg cell allowing one to use a small 6 cell neighborhood as done in [24]. Zero neighborhood size is a special case in dccrg signifying that only face neighbors of equal size are considered neighbors (with AMR all of the refined neighbor's children are considered instead). For example in a periodic grid without AMR neighborhood sizes of 1 and 2 would result in 26 and 124 neighbors per cell respectively.

Naturally the size of the neighborhood affects the amount of data that must be transferred between processes during remote neighbor data updates regardless of the grid implementation thus affecting parallel scalability. Additionally since in dccrg a maximum refinement level difference of one is enforced between neighboring cells, the size of the neighborhood does affect for example the amount of induced cell refinement (see Section 3.3).

3.1.4. Ease of use

Even though initially dccrg was developed only for in-house use, it was nevertheless designed to be simple to use for the kinds of simulations it is targeted for. Fig. 4 shows an example of a complete parallel program playing Conway's Game of Life using dccrg written in less than 60 lines of code (LOC) including whitespace

[2] https://gitorious.org/dccrg/dccrg/blobs/master/dccrg_constant_geometry.hpp

[3] User-defined data types in http://www.boost.org/doc/libs/release/doc/html/mpi/tutorial.html

```
1  #include "boost/foreach.hpp"
2  #include "boost/mpi.hpp"
3  #include "cstdlib"
4  #include "vector"
5  #include "zoltan.h"
6  #define DCCRG_CELL_DATA_SIZE_FROM_USER
7  #define DCCRG_USER_MPI_DATA_TYPE
8  #include "dccrg.hpp"
9
10 struct game_of_life_cell {
11    int data[2];
12    void* at() {return &(this->data[0]);}
13    MPI_Datatype mpi_datatype() {return MPI_INT;}
14 };
15
16 int main(int argc, char* argv[]) {
17    boost::mpi::environment env(argc, argv);
18    boost::mpi::communicator comm;
19
20    float zoltan_version;
21    Zoltan_Initialize(argc, argv, &zoltan_version);
22
23    dccrg::Dccrg<game_of_life_cell> grid;
24    grid.set_geometry(10, 10, 1, 0, 0, 0, 1.0, 1.0, 1.0);
25    grid.initialize(comm, "HYPERGRAPH", 1, 0);
26    grid.balance_load();
27    const std::vector<uint64_t> cells = grid.get_cells();
28
29    // initial state
30    BOOST_FOREACH(const uint64_t cell, cells) {
31       game_of_life_cell* cell_data = grid[cell];
32       cell_data->data[0] = cell_data->data[1] = 0;
33       if (cell % 4 == 0) cell_data->data[0] = 1;
34    }
35
36    for (int turn = 0; turn < 10; turn++) {
37       grid.update_remote_neighbor_data();
38
39       // collect live neighbor counts
40       BOOST_FOREACH(const uint64_t cell, cells) {
41          game_of_life_cell* cell_data = grid[cell];
42          cell_data->data[1] = 0;
43
44          const std::vector<uint64_t>* neighbors = grid.get_neighbors(cell);
45          BOOST_FOREACH(const uint64_t neighbor, *neighbors) {
46             if (neighbor == dccrg::error_cell) continue;
47             game_of_life_cell* neighbor_data = grid[neighbor];
48             if (neighbor_data->data[0] == 1) cell_data->data[1]++;
49          }
50       }
51       // assign new state
52       BOOST_FOREACH(const uint64_t cell, cells) {
53          game_of_life_cell* cell_data = grid[cell];
54          if (cell_data->data[1] == 3) cell_data->data[0] = 1;
55          else if (cell_data->data[1] != 2) cell_data->data[0] = 0;
56       }
57    }
58    return 0;
59 }
```

Fig. 4. A complete parallel program playing Conway's Game of Life using dccrg, see the text for details.

and comments. Lines 10–14 of the program define the class to be stored in every cell of dccrg along with member functions `at` and `mpi_datatype` which dccrg calls when querying the information required for transferring cell data between processes. The current state of a cell is saved into `data[0]` and the number of its live neighbors is saved into `data[1]`. On line 23 an instance of the grid is created with the class defined above as cell data. On line 24 the geometry of the grid is set to $10 \times 10 \times 1$ cells at refinement level 0 with minimum coordinate at (0, 0, 0) and cells of size 1 in each dimension. On line 25 the grid is initialized by setting the load balancing function to use in Zoltan, the neighborhood size and the maximum refinement level of cells. Lines 26 and 27 balance the load using Zoltan and collect the local cells. In this example the load is balanced only once and the local cell list does not change afterwards. On line 46 the non-existing neighbors of local cells are skipped. This is because the grid is initialized with non-periodic boundaries and neighbors that would be outside of the grid do not exist.

Fig. 5 shows relevant excerpts from a simple kinetic simulation showing the use of dccrg in the case of a variable amount of cell

data, the full program can be viewed in the dccrg git repository.[4] The remote neighbor update logic in the main simulation loop consists of the following steps:

1. The total number of particles in each cell is transferred between processes (lines 43 and 44).
2. Space for receiving particle data is allocated in local copies of remote cells based on their received total number of particles in step 1 (lines 47–52).
3. The particle coordinates are transferred between processes (lines 55 and 56).

The cell data class of the example kinetic simulation must provide the correct information to dccrg when updating remote neighbor data. The `at` and `mpi_datatype` functions now return a different address and number of bytes respectively depending on whether the total number of particles or the particle coordinates are transferred between processes. This is decided by the user in the main simulation loop. Additionally the `resize` function of the cell data class allocates memory for as many particles as there are in `number_of_particles`. A similar approach to the one described above is also used in our Vlasov simulation (further developed from [7]) where each real space cell has a separate adaptable velocity grid for ions consisting of a variable number of 4^3 cell velocity blocks.

In the previous example two communications are required per time step because processes receiving particle data do not know the number of incoming particles in advance. Since the MPI standard requires that the maximum amount of data to be received is known before calling the receive function the number of particles has to be communicated separately. This guarantees that processes receiving particles can specify the size of the data to MPI and allocate the memory required for received particles.

3.2. Load balancing/cell partitioning

Load balancing is also accomplished easily with dccrg. A user can call the `balance_load` function to let the Zoltan [26] library create a new partition, and single cells can also be moved manually between processes using the `pin` and `unpin` functions. Most of Zoltan's load balancing methods[5] can be used, namely: NONE, RANDOM, BLOCK, RCB, RIB, HSFC, GRAPH, HYPERGRAPH and HIER. In any case dccrg will transparently execute the new partition by transferring the necessary cell data between processes using MPI.

The structure of the grid in dccrg includes the owner of a cell in addition to the unique id of the cell (id is the key and owner is the value in a hash table). Thus the cell ids themselves are not used for partitioning cells between processes and any cell can be moved to any process (for example by using the RANDOM partitioner of Zoltan which we have found to be very useful for testing).

3.3. Adaptive mesh refinement algorithm

Due to the unique mapping of cells' ids and their physical location and size it is straightforward to refine any given cell in the grid, i.e. to calculate the ids of the children of any cell, and can be done locally (see Section 2). In order to enforce a maximum refinement level difference of one between neighboring cells whenever a cell is refined the refinement level of all neighbors is checked. If the refinement level of any neighbor is less than that of the cell being refined that neighbor is also refined. This is continued recursively until no more cells need to be refined. The size of the cells' neighborhood affects induced refinement indirectly by changing the number of neighbors a cell has and hence the potential number of cells whose refinement will be induced. In dccrg a few simplifications have been made to AMR: (1) any set of cells can only be refined once before calculating induced refinement (by `stop_refining`), e.g. induced refinement can only increase the refinement level of cells by one, and (2) unrefining a cell does not induce unrefinement, e.g. any cell which has at least one neighbor with refinement level larger than the cell being unrefined (i.e. it has a smaller neighbor) cannot be unrefined.

In a parallel setting the only difference to the above is that whenever a process refines or unrefines a cell that information has to be given to all processes which have cells within a certain distance of the cell that was refined or unrefined. Currently this distance is equal to infinity, e.g. all changes to the structure of the grid (refines and unrefines) are communicated globally. This has a significant impact on the parallel scalability of AMR in dccrg and is discussed in Section 4 and a method for making this minimum distance finite is discussed in Section 5. The changes in grid structure are exchanged between processes after each recursive step of induced refinement which are continued until no more cells need to be refined.

3.4. Parallel implementation

Good scalability in distributed memory machines requires asynchronous point-to-point MPI communication between processes with minimal total amount of communication, and especially minimal amount of global communication. The number of MPI messages should also be minimized in order not to burden the network with unnecessary traffic. Therefore any process must know which processes require data from local cells and from which remote cells data is required during remote neighbor updates without querying that information from other processes beforehand. Thus in dccrg every process knows the structure of the grid (e.g. which cells exist and which processes own them) to at least a certain distance from any of its cells so it can calculate locally which cells' data to send and receive from other processes. Due to this dccrg does not require global communication during ordinary time stepping, e.g. remote neighbor data updates, which enables excellent scalability when not doing load balancing or AMR as shown in Section 4.1. Internally dccrg precalculates these send and receive lists whenever the structure of the grid changes due to AMR or cells being moved between processes.

Even though the replicated mesh metadata of dccrg does not include the data stored in each cell it nevertheless limits the size of dccrg grids in practice to less than 100 M existing cells. This number does not depend on the refinement level or physical location of the cells and has so far been more than sufficient for our needs. To our knowledge the only parallel grid library that does not have any persistent global data structures is [20]. In [21,22] only the macrostructure of the grid (i.e. cells of refinement level 0) is known by all processes. According to the authors this limits the size of the grid to the order of $10^5 \ldots 10^6$ cells of refinement level 0 but does not limit the number of smaller cells.

4. Scalability results

The time stepping scalability of dccrg (e.g. without AMR or load balancing) depends mostly on the hardware running the simulation and on three parameters specific to each simulated system: (1) the time required to solve the inner cells of a process, (2) the amount of data transferred during the remote neighbor data update of a process and (3) the time required to solve the outer cells of a process. We show the dependency of dccrg scalability on these parameters by varying the total number of cells and processes and

[4] https://gitorious.org/dccrg/dccrg/blobs/master/tests/particles/simple.cpp.
[5] http://www.cs.sandia.gov/Zoltan/ug_html/ug_alg.html#LB_METHOD.

```
struct Cell {
    unsigned int number_of_particles;
    std::vector<boost::array<double, 3> > particles;
    static bool transfer_particles;

    // returns the starting address of data to send/receive
    void* at() {
        if (Cell::transfer_particles) {
            if (this->particles.size() > 0) {
                return &(this->particles[0]);
            } else {
                // return a sane address
                return &(this->number_of_particles);
            }
        } else {
            return &(this->number_of_particles);
        }
    }

    // returns the length in bytes to send/receive
    MPI_Datatype mpi_datatype() const {
        MPI_Datatype datatype;
        if (Cell::transfer_particles) {
            MPI_Type_contiguous(
                this->particles.size() * sizeof(boost::array<double, 3>),
                MPI_BYTE, &datatype);
        } else {
            MPI_Type_contiguous(1, MPI_UNSIGNED, &datatype);
        }
        return datatype;
    }

    void resize() {
        this->particles.resize(this->number_of_particles);
    }
}; // struct Cell

int main(...) {
    dccrg::Dccrg<Cell> grid;
    ...
    for (int step = 0; step < max_steps; step++) {
        // update number of particles between processes
        Cell::transfer_particles = false;
        grid.update_remote_neighbor_data();

        // allocate space for particles in copies of remote neighbors
        const boost::unordered_set<uint64_t>& remote_neighbors
            = grid.get_remote_cells_with_local_neighbors();
        BOOST_FOREACH(const uint64_t remote_neighbor, remote_neighbors) {
            Cell* data = grid[remote_neighbor];
            data->resize();
        }

        // update particle data between processes
        Cell::transfer_particles = true;
        grid.update_remote_neighbor_data();
    }
}
```

Fig. 5. Relevant excerpts from a simple kinetic simulation showing how to use dccrg when the amount of data in grid cells varies both in space and in time, see text for details.

by using a MHD solver in one, two and three dimensions. The run-time AMR scalability of dccrg is also presented.

The non-AMR scalability tests were carried out on three different supercomputers: (1) a 2 k core Cray XT5m system with 12 core nodes connected by SeaStar2 installed at the Finnish Meteorological Institute which we will refer to as Meteo, (2) a 295 k core IBM Blue Gene/P system with 128 core nodes (nodeboards) installed at the Jülich Supercomputing Centre which we will refer to as Jugene, and (3) a 12 k core bullx system with 32 core nodes connected by InfiniBand installed at the Très Grand Centre de Calcul which we will refer to as Curie.

4.1. MHD tests without AMR

First we show the time stepping scalability of dccrg in several MHD problems with a solver developed for the global MHD model GUMICS-4 [1] which solves ideal MHD equations in conservative form. Specifically the solver is a first order Roe's approximate

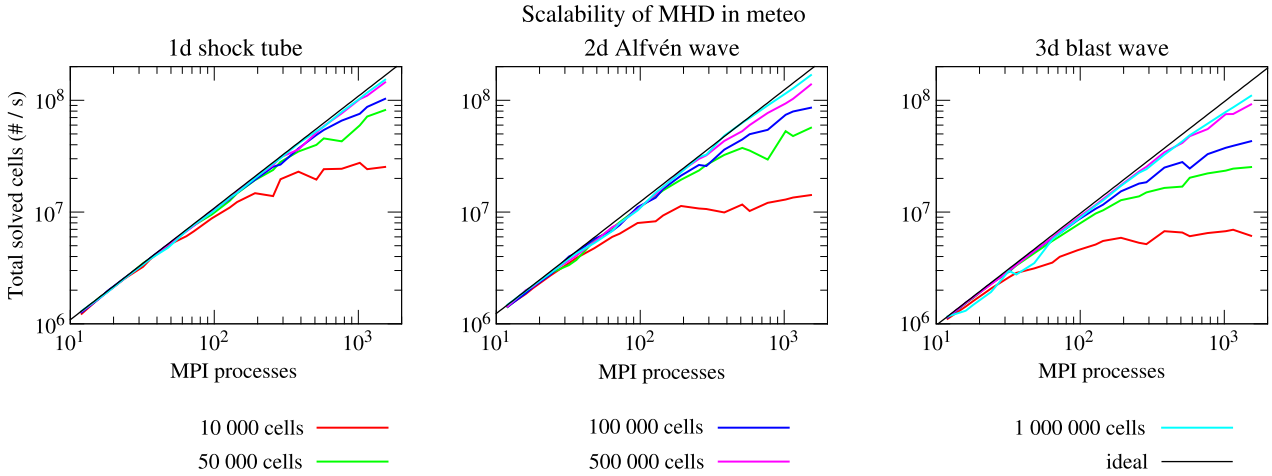

Fig. 6. Strong MHD scalability tests results with a static grid in one, two and three dimensions in Meteo (Cray XT5m). Total number of solved cells per second is shown as a function of the number of processes used and the total number of cells in each simulation. The ideal line is extrapolated from the 12 process result with 1 M cells.

Riemann solver for a Godunov type problem [27]. In the test results we present here only the Roe solver from GUMICS-4 is used as we do not experience problems with negative pressures or densities in the presented tests.

The nature of the solver is such that when solving the flux through a face data is only required from cells adjacent to the face, irrespective of the size of cells involved. Thus interpolation of data is not required at any point in the solution and if a cell has more than one face neighbor in any direction the flux through each common face is solved in the usual way. In the tests presented here we do not include a background magnetic field which is used in GUMICS-4 to represent the Earth's dipole field.

The problems we use are the one-dimensional shock tube presented for example in [28], the two-dimensional circularly polarized Alfvén wave presented by [29] and further elaborated on by [30], and the three-dimensional blast wave presented by [31]. Periodic boundary conditions are used in all tests, except for the shock tube test in the direction of the tube where initial conditions are enforced after every time step. In MHD tests every cell contains the cell-averaged values of the conservative MHD variables (density, momentum density, total energy density and magnetic field) giving a total of 128 bytes which must be transferred when updating the data of one cell between two processes. Since only the face neighbors of a cell are required for calculating the next time step we use a neighborhood size of zero in dccrg. In these tests processes execute one collective MPI communication per time step in order to dynamically calculate the maximum physical length of the time step. No other global communication is done. Since the grid is static in these tests the computational load is balanced only once at the start of the simulation by using a Hilbert space-filling curve[6] instead of Zoltan.

Fig. 6 shows the results of strong scalability tests using MHD with a static grid in Meteo in one, two and three dimensions. The total number of cells (10 k, 50 k, 0.1 M, 0.5 M and 1 M) is kept constant while the number of MPI processes is increased from 12 to 1536. In each test case scalability improves with the total number of cells because processes have more inner cells to solve while remote neighbor data is being transferred. For example in the shock tube test every process requires the data of two remote neighbors at most while the number of inner cells with 1.5 k processes increases from about 4 (10 k total cells) to 649 (1 M total cells). With 1 M cells the one and two dimensional tests scale almost ideally in Meteo and the three dimensional test is also quite close to ideal. The overall decrease in scalability with increasing number of dimensions is due to more data being transferred between processes for the same number of local cells.

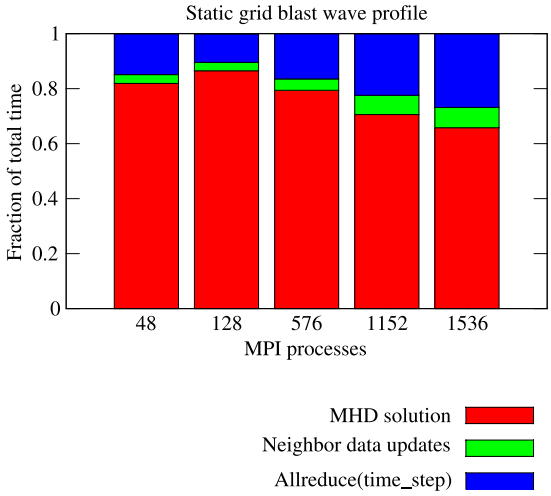

Fig. 7. Profile of the three dimensional MHD blast wave test without AMR using 1 M cells. Allreduce labels the only global communication executed each time step where the maximum length of the physical time step is obtained using `MPI_Allreduce`. Initialization and file I/O are not included.

As suggested by the scalability results above most of the simulation time is spent solving MHD which is shown in Fig. 7 for the three dimensional blast wave test using 1 M cells. The only global communication executed per time step while simulating is the calculation of the maximum length of the physical time step and is obtained using `MPI_Allreduce`. This is labeled as Allreduce in Fig. 7 and basically shows the computational and MPI imbalance between processes due to load balancing. Initialization and file I/O are not included in the profile and other parts of the simulation take an insignificant fraction of the total run time.

The non-AMR scalability tests were also carried out in Jugene and Curie and the results for the three-dimensional blast wave are shown in Figs. 8 and 9 respectively. Similarly to Meteo the one and two dimensional tests (not shown) scale better than the three-dimensional test in both Jugene and Curie. The overall results are similar in all tested machines, e.g. scalability improves with increasing number of total cells and decreasing number of dimensions. In Jugene very good scalability up to 32 k processes is obtained for a total number of cells of 1 M and above. The total

[6] https://gitorious.org/sfc/sfc/blobs/master/sfc++.hpp.

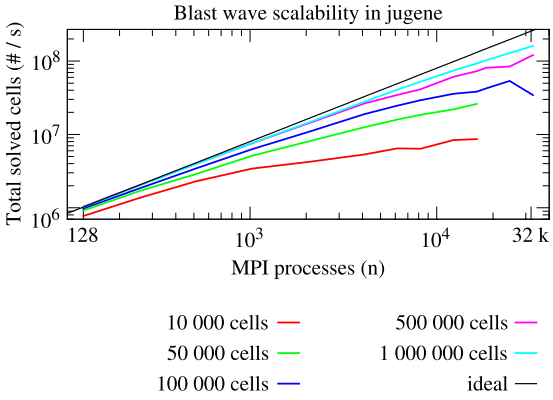

Fig. 8. Strong MHD scalability tests results with a static grid in three dimensions in Jugene (IBM Blue Gene/P). Total number of solved cells per second is shown as a function of the number of processes used and the total number of cells in each simulation. Ideal line is extrapolated from the 128 process result with 1 M cells.

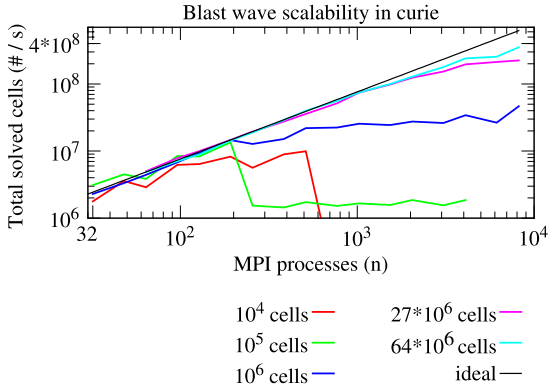

Fig. 9. Strong MHD scalability tests results with a static grid in three dimensions in Curie (bullx InfiniBand). Total number of solved cells per second is shown as a function of the number of processes used and the total number of cells in each simulation. Ideal line is extrapolated from the 32 process result with 64 M cells.

simulation speed in Jugene is only slightly above that of Meteo mostly due to the relatively small single-core performance of Jugene. Additionally the average number of cells per process is more than 20 times larger in Meteo than in Jugene for the maximum number of processes used but this has only a small effect on scalability in Jugene.

In Curie good scalability up to 8 k processes is obtained only with 64 M total cells but with a maximum solution speed of nearly 400 M solved cells per second which is over twice of that in Jugene. We attribute this to the relatively low node interconnect and high single-core and performance of Curie respectively when compared to Jugene.

4.2. Scalability of run-time AMR

Fig. 10 shows the speed of pure adaptive mesh refinement in dccrg. Initially the grid is 8^3, 16^3, 64^3 or 128^3 cells and every process refines all local cells until the total size of the grid is 128^3 or 256^3 cells. Initially the cells were partitioned using a space-filling curve and this is not included in the timings. Cells also were not transferred between processes during AMR. As can be seen in Fig. 10 the maximum cell refining speed of dccrg is of the order of 1 M cells per second. The linear scalability of AMR up to some 32 MPI processes is explained by the fact that after changes in the structure of the grid the arrow lists are recalculated only for local cells and MPI communication has not yet become a bottleneck. At 256 processes the amount of global communication required for updating the

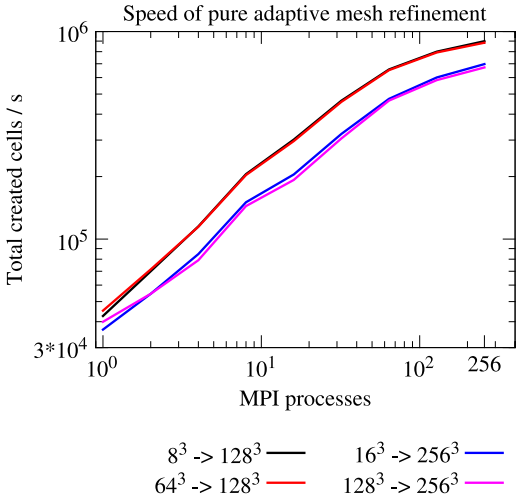

Fig. 10. Speed of adaptive mesh refinement in dccrg. Initial size of the grid is 8^3, 16^3, 64^3 or 128^3 cells. Every process refines each local cell until the total size of the grid is 128^3 or 256^3 cells.

structure of the grid between all processes starts to significantly affect the speed of AMR. This is discussed further in Section 4.3.

4.3. Scalability of blast wave test with AMR

Here we present the scalability of dccrg with AMR in the three-dimensional blast wave test used in Section 4.1. In this test a procedure similar to the one in GUMICS-4 (Eq. (2) in [1]) is used to decide whether to refine or unrefine a cell. A refinement index is calculated for each cell based on the relative difference of several variables between a cell and its face neighbors. Here the calculation of refinement index α additionally includes velocity shear relative to the maximum wave velocity from the cells' interface. The full equation for the refinement index α is:

$$\alpha = \max\left(\frac{\Delta\rho}{\widehat{\rho}}, \frac{\Delta U_1}{\widehat{U_1}}, \frac{(\Delta p)^2}{2\widehat{\rho U_1}}, \frac{(\Delta B_1)^2}{2\mu_0\widehat{U_1}}, \frac{|\Delta B_1|}{\widehat{B_1}}, \frac{(\Delta v)^2}{v_{\min}}\right)$$

where Δ denotes the difference in a variable between two cells, the hat denotes a minimum of the two values (as it actually does also in [1]), $v_{\min} = \widehat{v^2} + (0.01 \cdot v_{wave})^2$ and v_{wave} is the maximum wave velocity from the cells' interface. In this test a cell is refined if $\alpha > 0.02 \cdot (l+1)/L$, where l is the cell's current refinement level and $L = 4$ is the maximum refinement level of the grid. In other words a cell is refined to the maximum refinement level if its refinement index exceeds 0.02. A cell is unrefined if $\alpha < 0.02 \cdot (l+1)/L/2$, e.g. a cell is kept at refinement level 0 if $\alpha < 0.0025$ and none of the cell's neighbors' refinement levels exceed 1 (due to dccrg enforcing a maximum refinement level difference of one between neighbors).

We use a maximum refinement level of 4 in this test with an initial grid of 25^3 cells which results in an effective resolution of $400^3 = 64$ M cells. The computational load is balanced using Zoltan's recursive coordinate bisection (RCB) algorithm whenever the fraction of local cells ($f_c = N_{\max}/N_{\min}$, where max and min are the maximum and minimum number of local cells among all processes respectively) exceeded a specified limit. Animation 1 (Fig. 11 in the print version) shows from left to right, top to bottom the grid, pressure, magnetic and kinetic energy density during the simulation (at the end of the simulation in the print version) in the $y = 0$ plane when grid is adapted at every time step. At the end of the simulation the fraction of maximum to minimum values is 15 for density (not shown), 43 for pressure and 2.3 for magnetic energy density. Even though the MHD solver we use is simpler than the

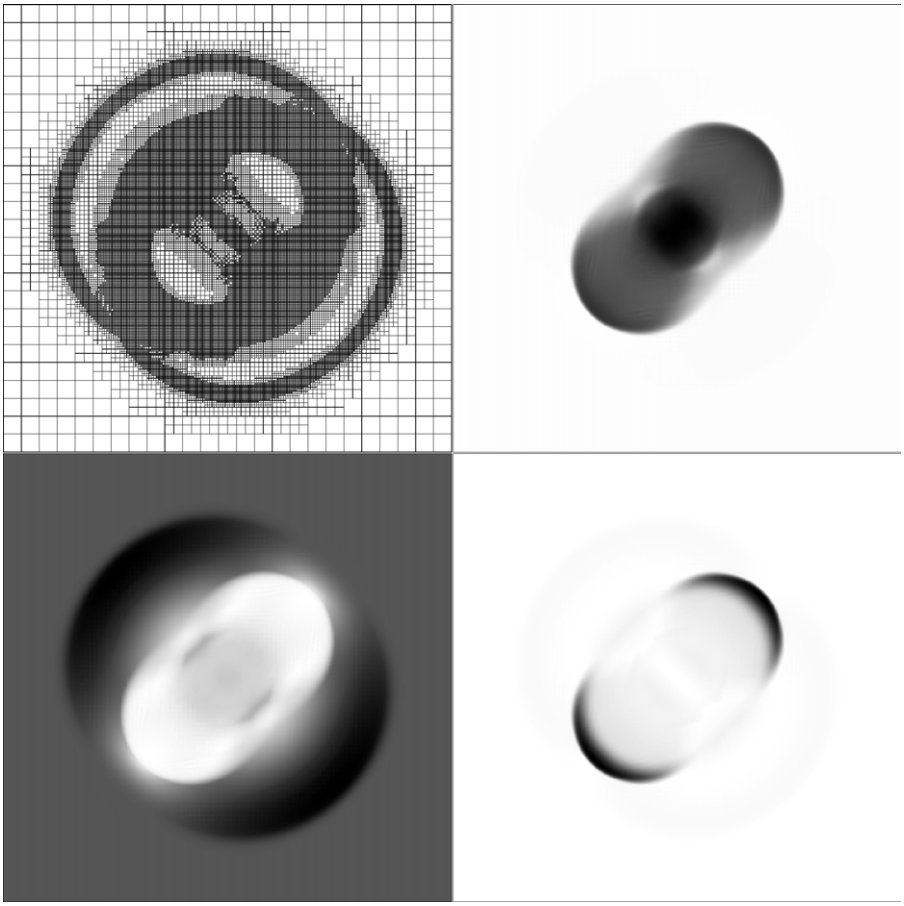

Fig. 11. Adaptive mesh refinement used in a MHD blast wave test showing from left to right, top to bottom the grid, pressure, magnetic and kinetic energy density during the simulation (final state of simulation in the print version) in the $y = 0$ plane when grid is adapted at every time step. At the end of the simulation the fraction of maximum to minimum values is 15 for density (not shown), 43 for pressure and 2.3 for magnetic energy density.
Source: From Ref. [31].

one in [31] the results are still close due to the high effective resolution achieved by using run-time AMR.

Fig. 12 shows the total solution speed during the simulations as a function of the number of MPI processes used. In the reference run a CFL [32] of 0.4 is used, AMR is done at every time step and the load is balanced whenever the local cell fraction $f_c \geq 2$. The AMR_N runs are otherwise identical to the reference run but CFL is set to $0.4/N$ and AMR is done every Nth time step, essentially multiplying the amount of non-AMR work in these runs by N. The results between different AMR runs are identical by visual inspection except for increased diffusion in runs with low CFL. The ratio of work required by AMR and the rest of the simulation has a significant effect on the total solution speed. The solution speed is a factor of 5 higher in the AMR_{32} run than in the reference run when using about 500 MPI processes. In the reference AMR run the total solution speed is close to 1/10 of the non-AMR version with up to 144 processes and in the AMR_{32} the speed is close to 1/3 with up to 288 processes after which both fractions start to decrease. We define these as the regions of excellent AMR scalability. On the other hand in all of the AMR runs the total solution speed increases up to about 500 to 600 processes after which it starts to decrease. We define this as the region where AMR is scalable. The total number of cells in the AMR runs averages to 4.5 M which is about 1/14 of the non-AMR version. Consequently in the region of excellent scalability the time to solution when using AMR is about 67% to 22% of the non-AMR time for the reference and AMR_{32} runs respectively. Even with a higher number of MPI processes it can still be advantageous to use AMR because the number of simulation cells is over a magnitude

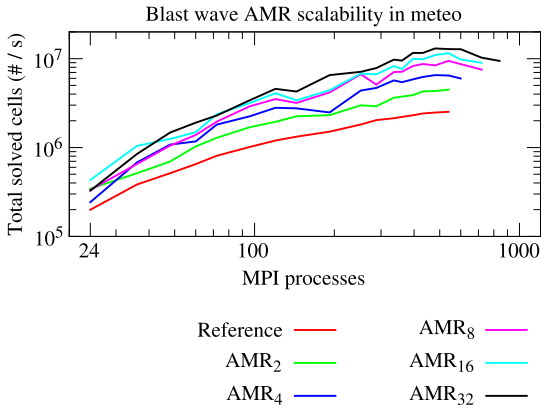

Fig. 12. Scalability of adaptive mesh refinement used in a MHD blast wave test [31]. In the reference run a CFL of 0.4 is used, AMR is done at every time step and the load is balanced whenever the local cell fraction f_c exceeded 2, where $f_c = N_{\max}/N_{\min}$ and max and min are the maximum and minimum number of local cells among processes respectively. The AMR_N runs are otherwise identical to the reference run but CFL is set to $0.4/N$ and AMR is done every Nth time step, essentially multiplying the amount of non-AMR work in these runs by N.

lower than without AMR. At the end of the AMR runs 9.9 M cells exist in the grid and the total number of cells created and removed is between 40.2 and 40.7 M, depending mostly on the diffusion, and averages to about 91 k added + removed cells per time step.

Fig. 13 shows which parts of the AMR blast wave test require the most time. As the number of processes is increased the largest

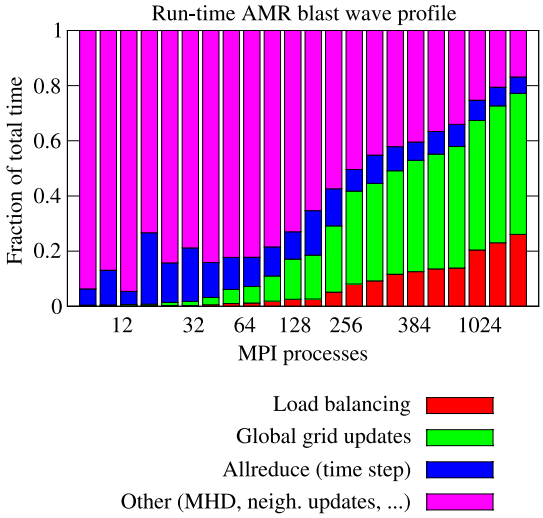

Fig. 13. Profile of the three dimensional MHD blast wave test with AMR using at most 10 M cells. Allreduce labels the calculation of the maximum length of the physical time step obtained using `MPI_Allreduce`. Initialization and file I/O are not included.

fraction of simulation time is spend in global communication related to AMR and load balancing. The Allreduce label again indicates global calculation of the physical time step and the Load balancing label indicates the simulation time spent in load balancing related functions. At about 300 MPI processes and above the largest fraction of simulation time is spent communicating changes in the structure of the grid between all processes. This includes both refining and unrefining cells as well as load balancing and in each case the `MPI_Allgatherv` function is used for distributing the changes in grid structure between all processes. Only a small fraction of the time spent in load balancing related functions is taken by Zoltan.

5. Discussion

While the ease of use of a software library is subjective it can be quantified by the number of lines of code required for usage and compared against other libraries when using the same programming language. With dccrg a complete parallel program playing Conway's Game of Life can be implemented in less than 60 LOC including whitespace and comments. Even though the required LOC is a crude estimate for a software library's ease of use it is nevertheless telling that such a short parallel program does not seem to be possible with other grid libraries.

The flexibility of dccrg also stands out since as far as we know only [25] allows one to easily exchange arbitrary cell data between MPI processes. Additionally dccrg supports transferring user-defined MPI datatypes which is critical for performance in some applications. For example when solving the 6 dimensional Vlasov equation in the Earth's magnetosphere [7] the simulation is heavily memory bound and using MPI datatypes directly for exchanging remote neighbor data is significantly faster than serializing said data into an additional buffer(s) before transfer. Dccrg also provides automatic neighbor data updates between processes with the ability of easily overlapping computation with communication.

Currently the largest drawback of dccrg is the fact that the entire structure of the grid is known by every process, i.e. a part of the mesh metadata is replicated by all processes. The global data structure prevents grids larger than about 100 M cells but this has not been a problem for us and can be worked around by storing blocks instead of single cells into dccrg (similarly to [14], for example). The global grid data structure of dccrg also reduces the scalability of AMR in the worst case to about 200 and overall to about 600 MPI processes. Nevertheless using AMR can lead to significant savings in the required memory as the number of cells can be one or even two orders of magnitude lower. Depending on the problem the required CPU time can also be significantly reduced when using AMR especially when the number of MPI processes used is of the order of 300 or less. It should also be noted that using threads to parallelize solvers within a shared memory node could effectively multiply the scalability range of simulations by the number of cores within one node, but this was not investigated.

Removing or significantly reducing the global data structure (as done in [20–22]) should improve both the largest attainable grid size and scalability of AMR considerably. Intuitively this is straightforward since with the exception of load balancing every process only needs to know the structure of the grid up to some finite distance from local cells. In order to be able to arbitrarily refine and unrefine grid cells without global communication local changes in the structure of the grid must be communicated between all neighboring processes. A neighboring process is defined as any process that has one or more of its cells inside the neighborhood of any cell of refinement level 0 that overlaps a local cell. In other words if only level 0 cells exist in the grid then the owners of all remote neighbors of local cells are considered as neighboring processes; and this holds no matter how the grid is subsequently refined and unrefined assuming that cells are not transferred between processes (load balancing). Global communication can also be avoided during load balancing if, for example, cells can be transferred only between neighboring processes. Even in this case new neighbor processes have to be recalculated but global communication is not required because cells could only have been transferred to/from a subset of all processes. Implementing this completely distributed mesh metadata is left to a subsequent study.

We presented the distributed cartesian cell-refinable grid (dccrg): an easy to use parallel structured grid library supporting adaptive mesh refinement and arbitrary C++ classes as cell data. Various MHD scalability results were presented and depending on the problem, hardware and whether AMR is used, excellent to average scalability is achieved. Dccrg is freely available for anyone to use, study and modify under version 3 of the GNU Lesser General Public License and can be downloaded from https://gitorious.org/dccrg.

Acknowledgments

This work is a part of the project 200141-QuESpace, funded by the European Research Council under the European Community's seventh framework programme. The research leading to these results has also received funding under grant agreement no. 260330 of the European Community's seventh framework programme. IH and MP are supported by project 218165 and AS is supported by project 251797 of the Academy of Finland. Results in this paper have in part been achieved using the PRACE Research Infrastructure resource Curie based in France at TGCC and Jugene based in Germany at JSC. IH thanks L.K.S. Daldorff, and J. Pomoell for insightful discussions.

Appendix. Supplementary data

Supplementary material related to this article can be found online at http://dx.doi.org/10.1016/j.cpc.2012.12.017.

References

[1] P. Janhunen, M. Palmroth, T. Laitinen, I. Honkonen, L. Juusola, G. Facskó, T.I. Pulkkinen, J. Atmos. Sol.-Terr. Phys. 80 (2012) 48 http://dx.doi.org/10.1016/j.jastp.2012.03.006.

[2] J.G. Lyon, J.A. Fedder, C.M. Mobarry, J. Atmos. Sol.-Terr. Phys. 66 (2004) 1333. http://dx.doi.org/10.1016/j.jastp.2004.03.020.
[3] K.G. Powell, P.L. Roe, T.J. Linde, T.I. Gombosi, D.L. De Zeeuw, J. Comput. Phys. 154 (1999) 284. http://dx.doi.org/10.1006/jcph.1999.6299.
[4] J. Raeder, R.J. Walker, M. Ashour-Abdalla, Geophys. Res. Lett. 22 (1995) 349.
[5] H.E.J. Koskinen, Physics of Space Storms: From the Solar Surface to the Earth, Springer-Verlag, 2011. http://dx.doi.org/10.1007/978-3-642-00319-6.
[6] E. Kallio, P. Janhunen, Ann. Geophys. 21 (2003) 2133 http://dx.doi.org/10.5194/angeo-21-2133-2003.
[7] M. Palmroth, I. Honkonen, A. Sandroos, Y. Kempf, S. von Alfthan, D. Pokhotelov, J. Atmos. Sol.-Terr. Phys. (2012) (in press) http://dx.doi.org/10.1016/j.jastp.2012.09.013.
[8] C.K. Birdsall, A.B. Langdon, Plasma Physics via Computer Simulation, McGraw-Hill Book Co., 1985.
[9] R.W. Hockney, J.W. Eastwood, Computer Simulation Using Particles, Adam Hilger, 1988.
[10] M.J. Berger, P. Colella, J. Comput. Phys. 82 (1989) 64 http://dx.doi.org/10.1016/0021-9991(89)90035-1.
[11] W.D. Henshaw, D.W. Schwendeman, J. Comput. Phys. 227 (2008) 7469. http://dx.doi.org/10.1016/j.jcp.2008.04.033.
[12] S.R. Kohn, S.B. Baden, J. Parallel Distrib. Comput. 61 (2001) 713 http://dx.doi.org/10.1006/jpdc.2001.1700.
[13] B.S. Kirk, J.W. Peterson, R.H. Stogner, G.F. Carey, Eng. Comput. 22 (2006) 237. http://dx.doi.org/10.1007/s00366-006-0049-3.
[14] B. van der Holst, R. Keppens, J. Comput. Phys. 226 (2007) 925 http://dx.doi.org/10.1016/j.jcp.2007.05.007.
[15] A.M. Khokhlov, J. Comput. Phys. 143 (1998) 519 http://dx.doi.org/10.1006/jcph.1998.9998.
[16] P. MacNeice, K.M. Olson, C. Mobarry, R. de Fainchtein, C. Packer, Comput. Phys. Comm. 126 (2000) 330. http://dx.doi.org/10.1016/S0010-4655(99)00501-9.
[17] Q.F. Stout, D.L. De Zeeuw, T.I. Gombosi, C.P.T. Groth, H.G. Marshall, K.G. Powell, Adaptive blocks: a high performance data structure, in: Proc. 1997 ACM/IEEE Conf. Supercomput. vol. 57, 1997. http://dx.doi.org/10.1109/SC.1997.10027.
[18] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, second ed., MIT Press and McGraw-Hill, 2001.
[19] B. Hariharan, S. Aluru, Parallel Comput. 31 (2005) 311. http://dx.doi.org/10.1016/j.parco.2004.12.007.
[20] H. Sundar, R.S. Sampath, G. Biros, SIAM J. Sci. Comput. 30 (2008) 2675. http://dx.doi.org/10.1137/070681727.
[21] C. Burstedde, L.C. Wilcox, O. Ghattas, SIAM J. Sci. Comput. 33 (2011) 1103. http://dx.doi.org/10.1137/100791634.
[22] W. Bangerth, C. Burstedde, T. Heister, M. Kronbichler, ACM Trans. Math. Software 38 (2011) 14:1–14:28. http://dx.doi.org/10.1145/2049673.2049678.
[23] M.S. Warren, J.K. Salmon, A parallel hashed oct-tree N-body algorithm, in: Proc. 1993 ACM/IEEE Conf. Supercomput., 1993. http://dx.doi.org/10.1145/169627.169640.
[24] R. Keppens, Z. Meliani, A.J. van Marle, P. Delmont, A. Vlasis, B. van der Holst, J. Comput. Phys. 231 (2012) 718. http://dx.doi.org/10.1016/j.jcp.2011.01.020.
[25] A.M. Wissink, R.D. Hornung, S.R. Kohn, S.S. Smith, N. Elliott, Large scale parallel structured AMR calculations using the SAMRAI framework, in: Proc. 2001 ACM/IEEE Conf. Supercomput., 2001. http://dx.doi.org/10.1145/582034.582040.
[26] K. Devine, E. Boman, R. Heaphy, B. Hendrickson, C. Vaughan, Comput. Sci. Eng. 4 (2002) 90. http://dx.doi.org/10.1109/5992.988653.
[27] P.L. Roe, J. Comput. Phys. 43 (1981) 357.
[28] M. Torrilhon, J. Plasma Phys. 69 (2002) 253 http://dx.doi.org/10.1017/S0022377803002186.
[29] G. Tóth, J. Comput. Phys. 161 (2000) 605 http://dx.doi.org/10.1006/jcph.2000.6519.
[30] T.A. Gardiner, J.M. Stone, J. Comput. Phys. 205 (2005) 509 http://dx.doi.org/10.1016/j.jcp.2004.11.016.
[31] T.A. Gardiner, J.M. Stone, J. Comput. Phys. 227 (2008) 4123 http://dx.doi.org/10.1016/j.jcp.2007.12.017.
[32] R. Courant, K. Friedrichs, H. Lewy, Math. Ann. 100 (1928) 32. http://dx.doi.org/10.1007/BF01448839 (in German); R. Courant, K. Friedrichs, H. Lewy, AEC Research and Development Report NYO-7689 (1956) 1.

Article II

Preliminary testing of global hybrid-Vlasov simulation: Magnetosheath and cusps under northward interplanetary magnetic field,
Journal of Atmospheric and Solar-Terrestrial Physics, volume 99, issue 0, 2013, pages 41-46.

Journal of Atmospheric and Solar-Terrestrial Physics 99 (2013) 41–46

Contents lists available at SciVerse ScienceDirect

Journal of Atmospheric and Solar-Terrestrial Physics

journal homepage: www.elsevier.com/locate/jastp

Preliminary testing of global hybrid-Vlasov simulation: Magnetosheath and cusps under northward interplanetary magnetic field

M. Palmroth [a,*], I. Honkonen [a,b], A. Sandroos [a], Y. Kempf [a,b], S. von Alfthan [a], D. Pokhotelov [a]

[a] Finnish Meteorological Institute, Helsinki, Finland
[b] University of Helsinki, Department of Physics, Helsinki, Finland

ARTICLE INFO

Article history:
Received 17 March 2012
Received in revised form
12 September 2012
Accepted 25 September 2012
Available online 18 October 2012

Keywords:
Magnetosphere
Global modelling
Magnetosheath
Cusp

ABSTRACT

Global magnetohydrodynamic (MHD) simulations have been successful in describing systems where the important spatial scales are larger than ion inertial length and the plasma has a well-defined temperature. The weakness of global one-fluid MHD simulations is their inability to model the multi-temperature, multi-component plasmas in the inner magnetosphere, where most of space-borne technology, including communication and navigation systems reside. We are developing a global hybrid-Vlasov simulation, where electrons are MHD fluid, but protons are modeled as distribution functions evolved in time using the Vlasov equation. This approach does not include the noise present in kinetic-hybrid simulations, but is computationally extremely challenging requiring petascale computations with thousands of cores. Here, we briefly review the status of our new parallel six-dimensional Vlasov solver. We carry out a test particle simulation and propagate the distribution functions using the electromagnetic fields of the GUMICS-4 global MHD simulation. Our main goal is to test the Vlasov solver in a global setup against the standalone GUMICS-4 global MHD simulation. The results shown here are obtained during due northward interplanetary magnetic field (IMF). We find that the magnetosheath and magnetopause plasma properties from the test particle simulation are in rough agreement with the results from the GUMICS-4 simulation. Furthermore, we show that the cusp injection patterns reproduce the expected behavior of northward IMF. The results indicate that our solver behaves sufficiently well, indicating that global hybrid-Vlasov simulations of this kind are feasible, promising improved global simulation capabilities in the future.

1. Introduction

As space missions provide measurements from a limited volume, theoretical and numerical models are important in understanding the observed variations. Depending on the problem, the equations to be solved range from kinetic (full particle orbit calculation) to hybrid (full ion dynamics, fluid electrons) and magnetohydrodynamic (fluid equations). At present MHD simulations are the only feasible tool to make large-scale simulations of the near-earth space in real time or near real time, allowing also space weather forecasts with L1 input data. Several global simulation groups have started to improve the accuracy of their codes by coupling the MHD part with inner magnetosphere models (e.g., Huang et al., 2006). Global (including solar wind, magnetosphere, and ionosphere) hybrid simulations in three dimensions exist for non-magnetized bodies (Kallio and Janhunen, 2002), while the first somewhat limited attempts for earth-based hybrid simulations have recently emerged (Omidi et al., 2005).

Ion energy dispersion with latitude across the polar cusp is an essential characteristic of cusp precipitations known to correlate with the interplanetary magnetic field (IMF) orientation. Under southward IMF conditions, near-equatorial dayside reconnection and subsequent poleward convection of the merged flux tubes causes a decreasing average energy of injected ions with increasing latitude as been proposed by Reiff et al. (1977). Under northward IMF conditions, lobe reconnection in the region poleward to the northern cusp results in a rising energy of injected particles at high latitudes causing the V-shaped dispersion, i.e. a decrease in energy followed by an increase as latitude increases (Woch and Lundin, 1992). Recent studies of cusp injections with a constellation of four Cluster satellites confirmed that the decreasing and the V-shaped energy-latitude dispersions are typical, respectively, for the southward and northward IMF conditions (Escoubet et al., 2008).

* Corresponding author. Tel.: +358 405311745.
E-mail address: minna.palmroth@fmi.fi (M. Palmroth).

1364-6826/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.jastp.2012.09.013

A number of MHD simulations were done to define the geometry of cusps under given geomagnetic conditions and to compare the results with spacecraft observations of cusp ion fluxes (Fenrich et al., 2001; Palmroth et al., 2001; Fuselier et al., 2002). However, the modeling of plasma entry into the cusps requires test particle tracing in the MHD simulation, as suggested by Fuselier et al. (2002), or more sophisticated treatment of ion dynamics such as Vlasov or kinetic simulations. Wang et al. (2009) also presented 3D hybrid simulations of cusp dynamics using kinetic treatment for ions and fluid treatment for electrons. Their study focused mainly on the sources of acceleration for cusp ions showing that the acceleration in the bow shock is the leading mechanism of ion energization, at least for the given simulation scenario.

We show here first preliminary testing of a new global hybrid-Vlasov simulation, Vlasiator, which is currently under development. We concentrate on the magnetosheath and cusp properties under northward IMF. This paper is organized as follows: First, we shortly present the two simulations. Second, we describe a simulation setup on February 18, 2004, during which the IMF conditions were almost due north, and compare the magnetosheath properties from the Vlasiator test-particle simulation to the standalone GUMICS-4 simulation. We show that results of the Vlasiator simulation are in rough agreement with the standalone GUMICS-4, suggesting that the solver is working sufficiently well. Furthermore, we show that the Vlasiator simulation reproduces the general patterns of the cusp injection under northward IMF conditions.

2. Models

2.1. GUMICS-4 global MHD simulation

GUMICS-4 (Janhunen et al., 2012) is a state-of-the-art global MHD simulation originally developed in 1996. GUMICS-4 solves the fully conservative MHD equations within the simulation box extending from $+32R_E$ to $-224R_E$ in the x direction and $\pm 64R_E$ in the yz directions. The inner shell of the magnetospheric domain is fixed at $3.7R_E$ distance, from where the magnetospheric domain is coupled with an electrostatic ionosphere. The magnetosphere determines the field-aligned currents and electron precipitation, which are given as boundary conditions to the ionospheric simulation domain. The field-aligned currents and the conductivity pattern resulting from precipitation and solar irradiation are used to determine the electric potential, which is given back to the magnetosphere, where it is used as an ionospheric boundary condition. Solar wind density, velocity, temperature and magnetic field are introduced as an input to the code at the sunward wall of the simulation box, while a variety of quantities are given as an output of the computation in space and time. GUMICS-4 uses adaptive mesh refinement, in which cells are hierarchically divided into eight smaller cells at regions with large spatial gradients.

2.2. Vlasiator

Vlasiator is a new simulation code that is currently in development. It is based on a hybrid-Vlasov model where protons are described by a full six-dimensional distribution function in the ordinary and velocity space obeying the Vlasov equation

$$\frac{\partial}{\partial t}f(\mathbf{r},\mathbf{v},t)+\mathbf{v}\cdot\nabla_r f(\mathbf{r},\mathbf{v},t)+\mathbf{a}\cdot\nabla_v f(\mathbf{r},\mathbf{v},t)=0, \tag{1}$$

where $\mathbf{r}$ and $\mathbf{v}$ are the ordinary space and velocity space coordinates, acceleration $\mathbf{a}$ is given by the Lorentz force, and $f(\mathbf{r},\mathbf{v},t)$ is the six-dimensional phase space density of protons with mass m and charge q. From this distribution function we can compute the ion charge density ρ_i and the ion current $\mathbf{j}_i$ as the zeroth and first moments of the velocity distribution by

$$\rho_i(\mathbf{r})=q\int f(\mathbf{r},\mathbf{v},t)\,d^3v \tag{2}$$

$$\mathbf{j}_i(\mathbf{r})=q\int \mathbf{v}f(\mathbf{r},\mathbf{v},t)\,d^3v. \tag{3}$$

From Eqs. (2) and (3) we can also compute the bulk speed of the protons as $\mathbf{V}_i=\mathbf{j}_i/\rho_i$.

In the hybrid-Vlasov model electrons are simulated as a charge neutralizing massless fluid, and not given a full Vlasov treatment, as that would be computationally difficult at the moment in global simulations. In this simulation the distribution functions were propagated in a test particle sense in electromagnetic fields that were obtained from GUMICS-4, i.e. the simulation was not self-consistent in this initial test of Vlasiator. Current version of Vlasiator is coupled to a field solver (Londrillo and del Zanna, 2004) making self-consistent simulations possible.

To describe the system, we use a three-dimensional Cartesian mesh in the ordinary space, which is parallelized using a domain decomposition scheme. Each cell in the ordinary space contains variables describing the electromagnetic field, as well as a three-dimensional velocity mesh describing the full six-dimensional phase-space. The distribution function is propagated forward in time with a finite volume method (FVM) using a high-order central scheme (Kurganov and Tadmor, 2000). In the FVM scheme the volume average $\tilde{f}$ is propagated forward in time by computing the fluxes at every cell face

$$\tilde{f}(t+\Delta t)=\tilde{f}(t)-\frac{\Delta t}{\Delta x}[F_x(x+\Delta x)-F_x(x)]-\frac{\Delta t}{\Delta y}[F_y(y+\Delta y)-F_y(y)] -\cdots-\frac{\Delta t}{\Delta v_z}[F_{vz}(v_z+\Delta v_z)-F_{vz}(v_z)], \tag{4}$$

where $\mathbf{F}_\mathbf{r}=(F_x,F_y,F_z)$ are the fluxes at cell faces in ordinary space and $\mathbf{F}_\mathbf{v}=(F_{vx},F_{vy},F_{vz})$ are the fluxes at cell faces in velocity space. These are given by

$$\mathbf{F}_\mathbf{r}=\mathbf{v}\tilde{f}, \tag{5}$$

$$\mathbf{F}_\mathbf{v}=\frac{q}{m}(\mathbf{E}+\mathbf{v}\times\mathbf{B})\tilde{f}. \tag{6}$$

Using Strang (1968) splitting we have split Eq. (4) into

$$\tilde{f}(\mathbf{r},\mathbf{v},t+\Delta t)=S_v\left(\frac{\Delta t}{2}\right)S_r(\Delta t)S_v\left(\frac{\Delta t}{2}\right)\tilde{f}(\mathbf{r},\mathbf{v},t), \tag{7}$$

where S_v is a three-dimensional propagator corresponding to the velocity space components of Eq. (4) and S_r is a three-dimensional operator corresponding to the ordinary space components.

3. Boundary and initial conditions for the test particle simulation

The simulation box in this version of Vlasiator extends from $+16R_E$ to $-24R_E$ in the x direction, and $\pm 20R_E$ in the yz directions. For every time step in this test particle simulation, the electromagnetic fields in the ordinary space are taken from the original GUMICS-4 simulation having a spatial resolution of $0.25R_E$. To save computation time, in the test particle simulation the fields are interpolated into $1R_E$ resolution, which describes the phase space for the distribution functions. Hence, in the test particle simulation the total number of ordinary space cells is 64,000. In velocity space we used 40^3 cells. For reference, we give some basic values of the simulation results in Table 1, showing

that while the simulation is still not reaching the full kinetic scales, the results will be helpful in validating the code and evaluating the code performance, and can be used as a reference in setting goals for the further development. Doubling the resolution in each coordinate direction would increase the simulation time by a factor of eight and the current simulation already took over a month to run on 768 CPU cores. Hence we leave the better resolution for further studies, and focus on outlining the main similarities and differences between Vlasov and MHD numerical solutions.

The GUMICS-4 simulation was carried out for an event observed on February 18, 2004. The simulated event consists of three substorms commencing at around 16 UT, 19 UT and 22:30 UT, with associated plasmoid formations and their subsequent release (Honkonen et al., 2011). While the Vlasiator time step was of the order of a millisecond, at every minute of the run the electromagnetic fields were taken from the GUMICS-4 simulation. In the Vlasiator simulation the distribution function density is initially zero everywhere. At each Vlasiator time step a Maxwellian distribution of protons with an average velocity of 450 km/s (according to the velocity observed during this event) was constantly inserted into the sunward edge of the test particle simulation, with outflow boundaries on the other edges. In this initial test there was no ionosphere, the boundary at earth was a simple outflow boundary condition at a $7R_E$ distance from the center of earth. This means that the distribution functions earthward of this shell are zero. Vlasiator is run for a physical time of 25 min, corresponding to 1.230,000 time steps, and therefore the distribution functions are only well-established within the solar wind and magnetosheath during the simulation, allowing to investigate the magnetosheath and cusp properties. The time period simulated with Vlasiator included a northward turning of the IMF, and the IMF during the time period investigated here in detail is strongly due north, $(0, -1, 9)$ nT, while the dipole tilt angle used throughout the run is -1°.

Table 1
Some basic plasma parameters in the simulation given in the solar wind, magnetosheath (0, 0, $25R_E$) and north lobe (-20, 0, $15R_E$), taken from the GUMICS-4 simulation.

Parameter	Solar wind	Magnetosheath	North lobe
Plasma beta	0.15	2	0.02
Proton Larmor radius (km)	65	174	247
Proton inertial length (km)	133	100	1440
Plasma frequency (kHz)	15	21	1.5
Lower hybrid frequency (Hz)	5.7	9.5	9.3

4. Results

Fig. 1 shows a comparison between the GUMICS-4 and Vlasiator densities in the noon–midnight meridian plane. Color-coding shows the plasma density; for Vlasiator we have calculated moments of the distribution function to facilitate comparison with GUMICS-4. The Vlasiator density corresponds quite well to GUMICS-4 MHD solution on the dayside. For example, the plasma density in the bow shock from the two simulations is quite close to each other even though initially the density in the Vlasiator simulation domain is zero. We consider this an extremely good result, keeping in mind that the resolution in the bow shock region is two times larger in standalone GUMICS-4. Furthermore, Fig. 1 also displays the velocity in the *xz* plane as arrows. The large Vlasiator velocities at the inner shell of magnetospheric domain are due to the zeroing and hence are not to be compared, instead we concentrate on the general features of the modeling results.

Fig. 1a and b shows three trajectories on which the densities are given in panels c–e for GUMICS-4 (red) and Vlasiator (blue). The zero point for the trajectories is the center of the Earth. As can be seen, the outer edge of the density increase due to bow shock appears in the same location, while GUMICS-4 shows slightly larger densities, the magnitude of the density is roughly the same in both simulations. There are also differences: due to the presence of ionospheric outflows, GUMICS-4 shows larger densities around $10R_E$ distance along the top and bottom trajectories. In Vlasiator the ionospheric outflow is currently not included.

The most notable difference between the model results in Fig. 1a and b is the magnetosheath velocity field, which in GUMICS-4 indicates a normal MHD-like behavior, where the solar wind flow impinges the earth's magnetic field and deviates to both lobes quite steadily. However in the Vlasiator, the flow trajectories within the magnetosheath are different. The flows above the ecliptic plane behave roughly as in GUMICS but below the ecliptic plane the flows continue tailward and then deflect towards the northern hemisphere. The flow deflection point in this plane occurs roughly at $(3,0,-15)R_E$, i.e. near the southern lobe reconnection region, which was originally identified from the

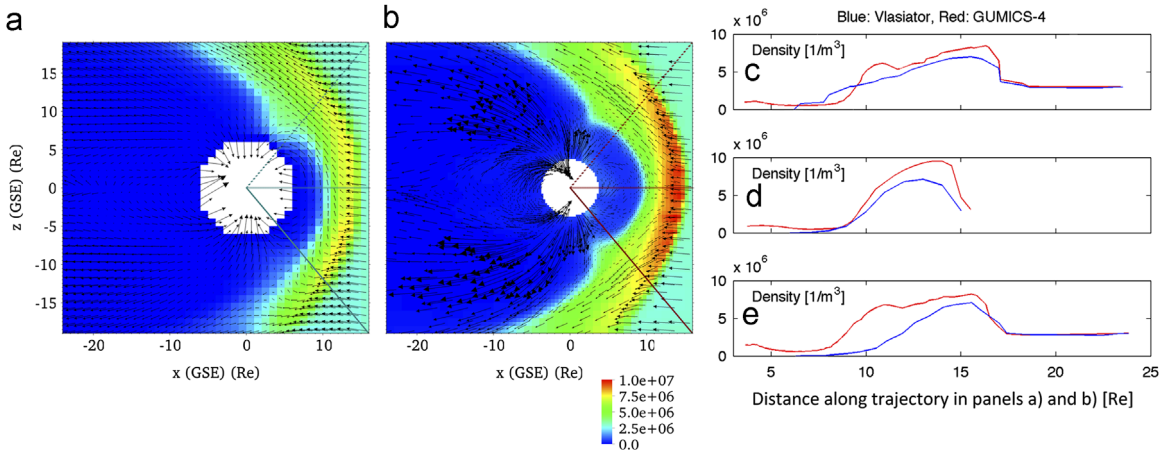

Fig. 1. Density in the GSE noon–midnight meridional plane in (a) Vlasiator, and in (b) GUMICS-4. The small arrows give the in-plane velocity field. Density along the three trajectories in panels a and b are given in panels c–e, with Vlasiator densities colored blue and GUMICS densities with red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

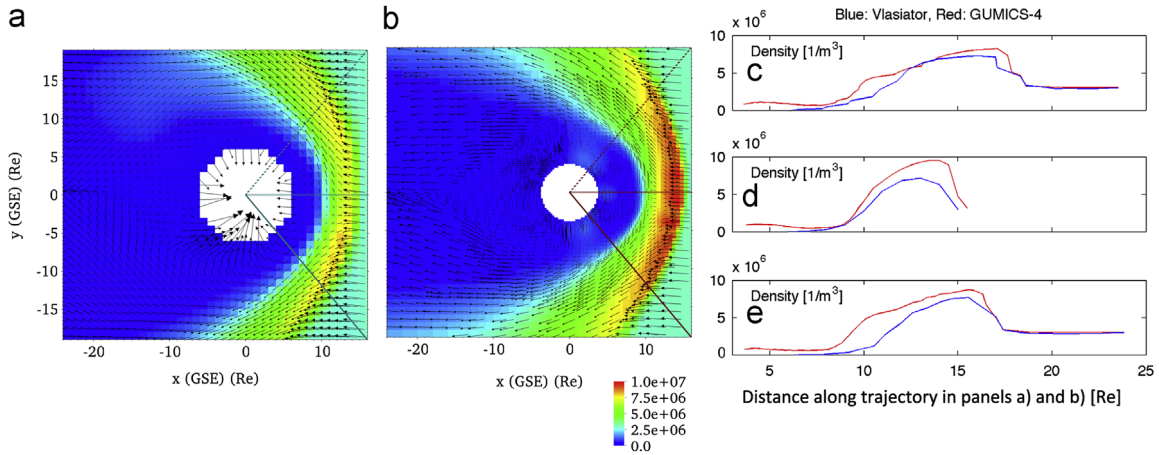

Fig. 2. Density in the GSE equatorial plane in (a) Vlasiator, and in (b) GUMICS-4. The small arrows give the in-plane velocity field. Density along the three trajectories in panels a and b are given in panels c–e, with Vlasiator densities colored blue and GUMICS densities with red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

GUMICS-4 simulation. This flow deflection also transports plasma towards the northern hemisphere, and as a result the southern magnetosheath is narrow and the northern magnetosheath broader, as can also be seen by comparing the densities in both models along the top and middle trajectories in Fig. 1c and e.

Fig. 2 shows the densities in the equatorial plane in a similar format as in Fig. 1. GUMICS-4 shows slightly larger values for density, but the density increases at the outer edge of the bow shock in roughly the same location in both models. Vlasiator shows lower densities within the magnetosheath near the magnetopause especially within the dusk-side magnetosheath. On the dawn side, the magnetosheath is broader and the density values are closer to those in GUMICS-4. The velocity flow in GUMICS-4 is MHD-like and deflects steadily to both sides of the magnetopause. In Vlasiator, the flow is more symmetric with respect to the dawn and dusk than compared to north and south in Fig. 1. However, the flow reversal and the stagnation point occurs slightly more within the dawn than in the dusk.

Panels a–c in Fig. 3 illustrate the plasma velocity in GUMICS-4 (red), and in Vlasiator (blue) along the trajectories in Fig. 1a and b, while panels d–f show velocities along trajectories in Fig. 2a and b. The Vlasiator velocity is the moment of the distribution function to facilitate comparison with GUMICS-4. The velocities in the equatorial plane (Fig. 3, panels d–f) are in good correspondence with GUMICS-4 elsewhere except near the start of the trajectory, where the inner shell of the magnetospheric domain influences the model results. Within the region where good comparison is to be expected, Vlasiator has slightly larger values on the morning side at around $10R_E$ (Fig. 3, panel d), otherwise the velocities are in quantitative agreement. Larger differences are observed in Fig. 3, panels a–c, showing the model results in the noon–midnight meridian plane.

Fig. 4a and b illustrates the $\mathbf{E} \times \mathbf{B}$ drift magnitude along the velocity field lines, above the northern and southern cusps, respectively, while Fig. 4c and d depicts the field-aligned velocity component along those same field lines in the Vlasiator simulation. In the southern hemisphere, the velocity field lines first go around the cusp and then change their path due to the newly established reconnection in the southern lobe. The color-coding illustrates that in the southern hemisphere the reconnection deviates the protons both into the southern cusp as well as towards the northern hemisphere causing the flow pattern visible in Fig. 1. In the northern hemisphere the protons drift into the cusp, and cause the larger density moments near the northern cusp in Fig. 1a.

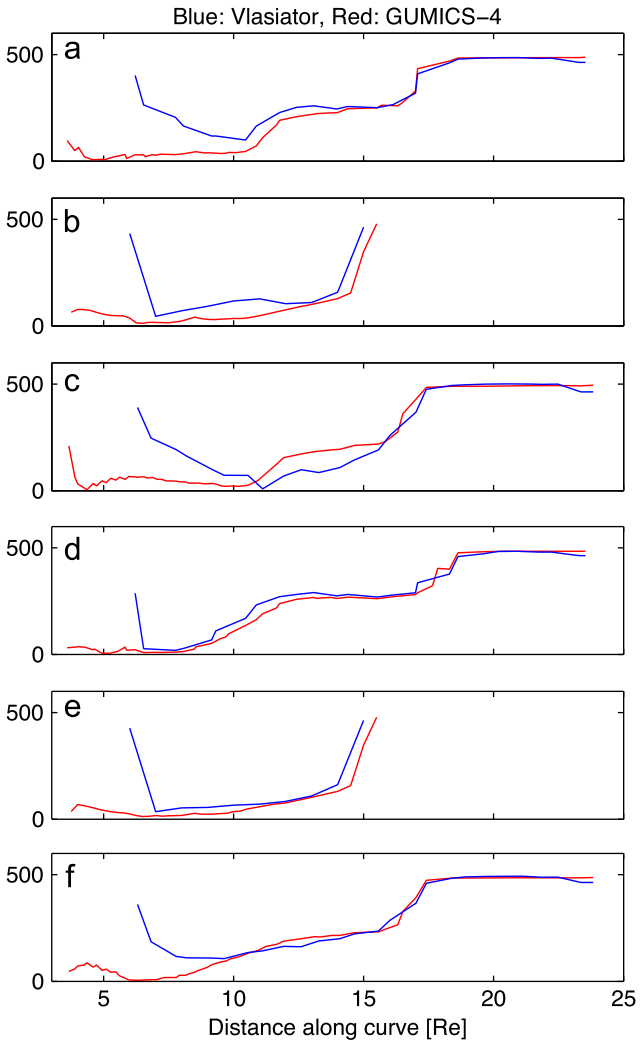

Fig. 3. Velocities in the (a)–(c) *xz* plane and (d)–(f) *xy* plane in GUMICS (red) and in Vlasiator (blue) along the trajectories in Figs. 1a and b and 2a and b, respectively. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

In Fig. 5 we illustrate the ion energy dispersion in the northern hemisphere by showing ion velocity profiles across the simulation domain at selected radial distances. Fig. 5 presents bulk ion

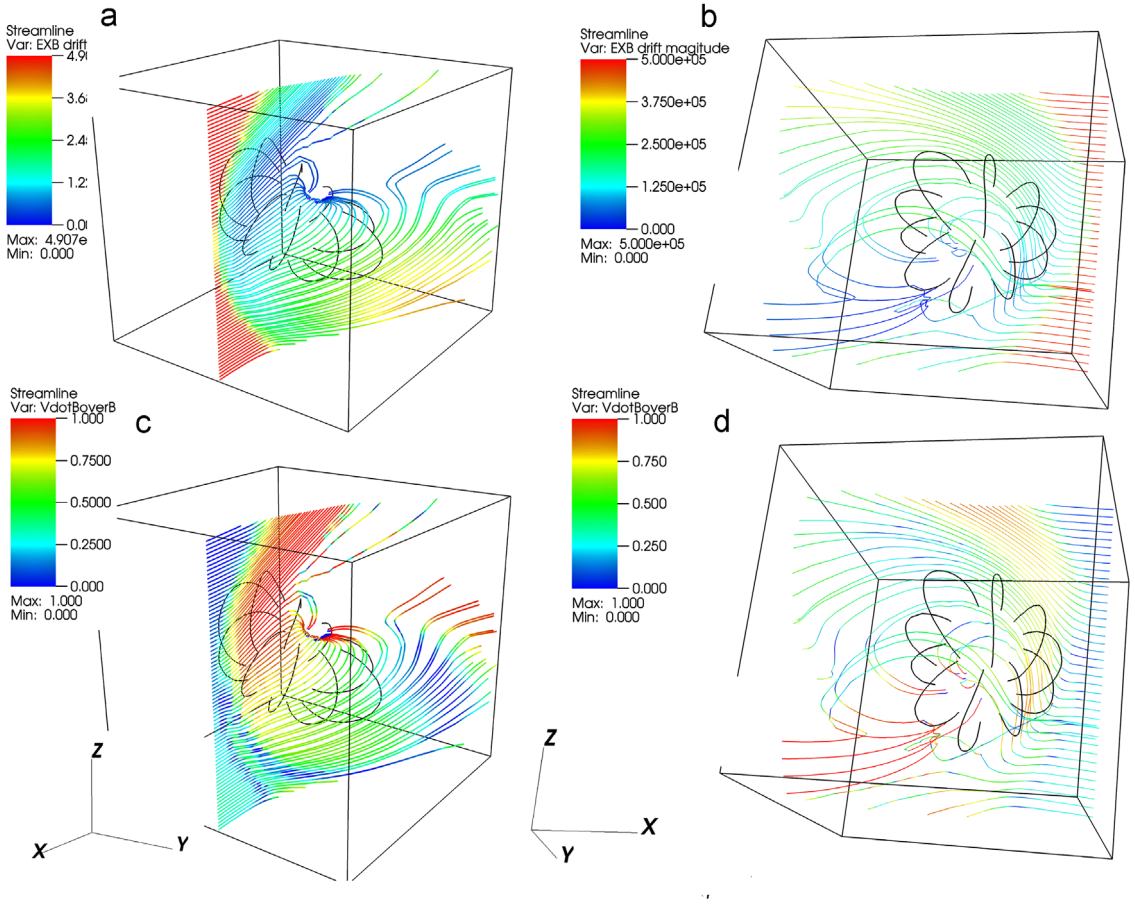

Fig. 4. The $\mathbf{E} \times \mathbf{B}$ drift magnitude color-coded on the velocity field lines (all components) above the northern (a) and southern (b) cusps. On the bottom panels is the fraction of the velocity aligned to the magnetic field color-coded on the same velocity field lines above northern (c) and southern (d) cusps. The viewing angle is indicated with the GSE coordinate axes in the bottom of the plots. The black curves are magnetic field lines to indicate the position of the Earth's dipole field. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

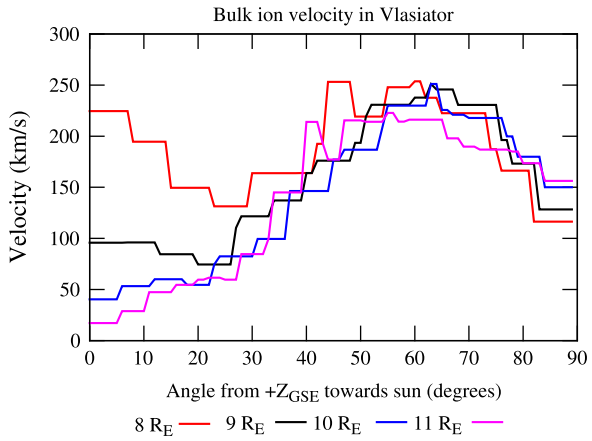

Fig. 5. Ion bulk velocity from Vlasiator at radial distances of 8, 9, 10 and $11R_E$ as a function of the angle away from the z_{GSE} axis towards the Sun.

velocity calculated from Vlasiator at radial distances of 8, 9, 10 and $11R_E$ as a function of angle away from the $+z_{GSE}$ axis towards the sun. A tilt of about 30° towards $+y_{GSE}$ was also used above in order to account for the dipole tilt in the GUMICS-4 simulation. The expected V-shaped dispersion is most pronounced at smaller radial distances ($8-9R_E$) while at larger radial distances the ion flow is dominated by anti-sunward $\mathbf{E} \times \mathbf{B}$ drift. In these simulations the V-shaped latitude dispersion arises naturally from the global field topology with two energetic ion streams arriving from the southern lobe reconnection equatorward of the cusp and from northern lobe reconnection poleward of the cusp.

5. Discussion

In this paper we have shown results from the initial comparison of test-particle Vlasiator simulation against the GUMICS-4 MHD simulation, concentrating on the northward IMF and magnetosheath and cusp properties. Most importantly, we demonstrate that a global six-dimensional hybrid-Vlasov simulation is possible to carry out with good parallelization with today's supercomputers. Secondly, we show that the solver works reasonably well so that the results can be compared to a state-of-the art simulation. Generally speaking, plasma properties within the two simulations are in rough agreement in the magnetosheath and cusp, where the Vlasiator distribution functions are well established during the 25 min simulation period. In Vlasiator the magnetosheath density is slightly smaller than in GUMICS-4, while the velocity appears to be larger. This might be due to the mass continuity and Vlasiator densities are smaller simply because the velocity is larger. Additionally, the ionospheric outflow in GUMICS explains differences between the simulation results near the low-altitude cusps.

The location of the Vlasiator bow shock is roughly the same as in GUMICS-4, although there are slight asymmetries in the thickness of the magnetosheath in northern and southern hemisphere in the Vlasiator simulation results. The origin of the asymmetry may be explained in two ways: (1) Either the ionospheric outflow present in GUMICS-4 but absent in Vlasiator may influence the results or (2) the newly established southern lobe reconnection may have consequences in the Vlasiator due to the kinetic treatment of the plasma that is not present in GUMICS-4 MHD simulation. In the second scenario, the establishing lobe reconnection accelerates magnetic field lines towards dayside and towards the northern hemisphere, and results in extra velocity in the phase space of Vlasiator and leads to an asymmetric flow pattern and thinner (thicker) magnetosheath in the southern (northern) side. Since GUMICS-4 describes the plasma in a fluid-like fashion, the magnetosheath flows in the GUMICS-4 magnetosheath are quite MHD-like and steadily divert into both hemispheres, leading to the difference between the simulation results. However, with the present resolution of Vlasiator we cannot decisively judge between the suggested scenarios and hence we leave this as a subject of a further study.

We also attempted to reproduce known features of the energy-latitude dispersion in cusp precipitations that are not present in global MHD simulations. Vlasiator shows a V-shaped pattern in the ion velocity profile across the polar cusp which corresponds to the energy-latitude dispersion typically observed under northward IMF conditions. These features of the cusp ion flow are absent in the GUMICS-4 simulation. Obviously the analysis of latitude dispersion in these simulations is only illustrative and does not intend to reproduce the actual cusp energy-latitude dispersion in details but is rather the first attempt in this direction. Real magnetosheath flow would have broader and more complex energy/velocity ion distributions and of course higher spatial resolutions, and hence other simulations will be needed to resolve the cusp plasma injections in more detail.

In summary, we find that the Vlasiator code reproduces the general pattern of the density distribution within the magnetosheath and exhibits a energy-latitude dispersion within the cusp. Furthermore, the results show that a global six-dimensional Vlasov code will be possible to be run with the near-future computing power.

References

Escoubet, C.P., Berchem, J., Bosqued, J.M., Trattner, K.J., Taylor, M.G.G.T., Pitout, F., Laakso, H., Masson, A., Dunlop, M., Dandouras, I., Reme, H., Fazakerley, A.N., Daly, P., 2008. Effect of a northward turning of the interplanetary magnetic field on cusp precipitation as observed by cluster. Journal of Geophysical Research 113, A07S13, http://dx.doi.org/10.1029/2007JA012771.

Fenrich, F.R., Luhmann, J.G., Fedder, J.A., Slinker, S.P., Russell, C.T., 2001. A global MHD and empirical magnetic field model investigation of the magnetospheric cusp. Journal of Geophysical Research 106, 18789–18802, http://dx.doi.org/10.1029/2001JA900040.

Fuselier, S.A., Berchem, J., Trattner, K.J., Friedel, R., 2002. Tracing ions in the cusp and low-latitude boundary layer using multispacecraft observations and a global MHD simulation. Journal of Geophysical Research 107 (A9), 1226, http://dx.doi.org/10.1029/2001JA000130.

Honkonen, I., Palmroth, M., Pulkkinen, T.I., Janhunen, P., Aikio, A., 2011. On large plasmoid formation in a global magnetohydrodynamic simulation. Annals of Geophysics 29, 167–179.

Huang, C.-L., Spence, H.E., Lyon, J.G., Toffoletto, F.R., Singer, H.J., Sazykin, S., 2006. Storm-time configuration of the inner magnetosphere: Lyon–Fedder–Mobarry MHD code, Tsyganenko model, and GOES observations. Journal of Geophysical Research 111, A11S16, http://dx.doi.org/10.1029/2006JA011626.

Janhunen, P., Palmroth, M., Laitinen, T.V., Honkonen, I., Juusola, L., Facskó, G., Pulkkinen, T.I., 2012. The GUMICS-4 global MHD magnetosphere – ionosphere coupling simulation. Journal of Atmospheric and Solar–Terrestrial Physics 80, 48–59. http://dx.doi.org/10.1016/j.jastp.2012.03.006.

Kallio, E., Janhunen, P., 2002. Ion escape from Mars in a quasineutral hybrid model. Journal of Geophysical Research 107 (A3), 1035, http://dx.doi.org/10.1029/2001JA0000902002.

Kurganov, A., Tadmor, E., 2000. New high-resolution central schemes for nonlinear conservation laws and convection–diffusion equations. Journal of Computational Physics 160, 241–282.

Londrillo, P., del Zanna, L., 2004. On the divergence-free condition in Godunov-type schemes for ideal magnetohydrodynamics: the upwind constrained transport method. Journal of Computational Physics 195.

Omidi, N., Blanco-Cano, X., Russell, C.T., 2005. Macrostructure of collisionless bow shocks: 1. Scale lengths. Journal of Geophysical Research 110, A12212, http://dx.doi.org/10.1029/2005JA011169.

Palmroth, M., Janhunen, P., Pulkkinen, T.I., Peterson, W.K., 2001. Cusp and magnetopause locations in global MHD simulation. Journal of Geophysical Research 106, 29435–29450, http://dx.doi.org/10.1029/2001JA900132.

Reiff, P.H., Burch, J.L., Hill, T.W., 1977. Solar wind plasma injection at the dayside magnetospheric cusp. Journal of Geophysical Research 82, 479–491, http://dx.doi.org/10.1029/JA082i004p00479.

Strang, G., 1968. On the construction and comparison of difference schemes. SIAM Journal on Numerical Analysis 5 (3), 506–517.

Wang, X.Y., Lin, Y., Chang, S.-W., 2009. Hybrid simulation of foreshock waves and ion spectra and their linkage to cusp energetic ions. Journal of Geophysical Research 114, A06203, http://dx.doi.org/10.1029/2008JA013745.

Woch, J., Lundin, R., 1992. Magnetosheath plasma precipitation in the polar cusp and its control by the interplanetary magnetic field. Journal of Geophysical Research 97, 1421–1430, http://dx.doi.org/10.1029/91JA02487.

Article III

On the performance of global magnetohydrodynamic models in the Earth's magnetosphere,
Space Weather, volume 11, issue 5, 2013, pages 313-326.

SPACE WEATHER, VOL. 11, 313–326, doi:10.1002/swe.20055, 2013

On the performance of global magnetohydrodynamic models in the Earth's magnetosphere

I. Honkonen,[1,2] L. Rastätter,[3] A. Grocott,[4] A. Pulkkinen,[3,5] M. Palmroth,[1] J. Raeder,[6] A. J. Ridley,[7] and M. Wiltberger[8]

Received 13 February 2013; revised 25 April 2013; accepted 28 April 2013; published 30 May 2013.

[1] We study the performance of four magnetohydrodynamic models (BATS-R-US, GUMICS, LFM, OpenGGCM) in the Earth's magnetosphere. Using the Community Coordinated Modeling Center's Run-on-Request system, we compare model predictions with magnetic field measurements of the Cluster, Geotail and Wind spacecraft during a multiple substorm event. We also compare model cross polar cap potential results to those obtained from the Super Dual Auroral Radar Network (SuperDARN) and the model magnetopause standoff distances to an empirical magnetopause model. The correlation coefficient (CC) and prediction efficiency (PE) metrics are used to objectively evaluate model performance quantitatively. For all four models, the best performance outside geosynchronous orbit is found on the dayside. Generally, the performance of models decreases steadily downstream from the Earth. On the dayside most CCs are above 0.5 with CCs for B_x and B_z close to 0.9 for three out of four models. In the magnetotail at a distance of about –130 Earth radii from Earth, the prediction efficiency of all models is below that of using an average value for the prediction with the exception of B_z. B_x is most often best predicted and correlated both on the dayside and the nightside close to the Earth whereas in the far tail the CC and PE for B_z are substantially higher than other components in all models. We also find that increasing the resolution or coupling an additional physics module does not automatically increase the model performance in the magnetosphere.

Citation: Honkonen, I., L. Rastätter, A. Grocott, A. Pulkkinen, M. Palmroth, J. Raeder, A. J. Ridley, and M. Wiltberger (2013), On the performance of global magnetohydrodynamic models in the Earth's magnetosphere, *Space Weather*, *11*, 313–326, doi:10.1002/swe.20055.

1. Introduction

[2] The growing interest in space weather forecasting from both government and industry is also increasing the need for space weather model development. The increasing amount of infrastructure and people that can be affected by severe space weather events demand reliable forecasting of those events and their effects both in space and on the ground. This in turn requires systematic testing, verification and validation of space weather models and the objective evaluation of their suitability for a particular purpose. The international need for model improvement and validation was highlighted, for example, in the recent European Commission's Space Weather Awareness Dialogue [*Krausmann and Bothmer*, 2012].

[3] A good overview of the process of verifying and validating a space plasma model is given by *Ledvina et al.* [2008]. The verification of a code consists of, for example, comparing the results to an analytic solution (e.g., using

[1]Finnish Meteorological Institute, Helsinki, Finland.
[2]Department of Physics, University of Helsinki, Helsinki, Finland.
[3]Community Coordinated Modeling Center, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA.
[4]Department of Physics and Astronomy, University of Leicester, Leicester, UK.
[5]Institute for Astrophysics and Computational Sciences, Catholic University, Washington, DC, USA.
[6]Space Science Center and Physics Department, University of New Hampshire, Durham, New Hampshire, USA.
[7]Department of Atmospheric, Oceanic and Space Sciences, University of Michigan, Ann Arbor, Michigan, USA.
[8]High Altitude Observatory, National Center for Atmospheric Research, Boulder, Colorado, USA.

Corresponding author: I. Honkonen, Earth Observation, Finnish Meteorological Institute, PL 503, 00101, Helsinki, Finland. (ilja.honkonen@fmi.fi)

1542-7390/13/10.1002/swe.20055

the method of manufactured solutions employed recently by *Welling et al.* [2012] for verifying SpacePy); monitoring conserved quantities, symmetries, and other predictable outcomes; or comparing results to those from other codes. Verification must happen before validation in order to make sure that the equations chosen for modeling the system are being solved correctly, and the results should be published especially in the case of a new model or a scheme. In this regard global magnetohydrodynamic (MHD) models leave something to be desired as only the BATS-R-US (Block Adaptive Tree Solar-wind Roe Upwind Scheme) code has been verified against analytic or semi-analytic results in a publication [*Powell et al.*, 1999]. Results for the ubiquitous shock tube test (see *Ryu and Jones* [1995] for a plethora of examples) have not been presented for the MHD solver(s) of any global MHD model even though such tests have been conducted for all models used here. For example, the MHD solvers of GUMICS were used in several one, two, and three-dimensional tests by *Honkonen et al.* [2013], but in addition to the parallel scalability results of all tests, only the result of a three-dimensional blast wave test with adaptive mesh refinement (AMR) was shown. The validation of a code can consist of controlled experiments designed to investigate a physical process, experiments specifically designed to validate codes or passive observations of physical events. Global MHD models have mostly been validated with many qualitative comparisons against observations by various spacecraft and, to our knowledge, no systematic quantitative comparisons have been conducted until recently.

[4] The Geospace Environment Modeling (GEM) 2008–2009 challenge represents the largest effort to date to validate global MHD models against observations using various objective metrics. *Pulkkinen et al.* [2011] quantified the performance of three global MHD models and two empirical models in reproducing observations of ground magnetic field during four geospace storm events. The models were compared against observations from 12 observatories located between 43.5° and 74° of geomagnetic latitude. Five different metrics, each applicable to different situations, were used to evaluate the model performance: root-mean-square difference, prediction efficiency, log-spectral distance, utility, and ratio of maximum amplitudes. *Rastätter et al.* [2011] used the same events and global MHD models as *Pulkkinen et al.* [2011] along with two empirical magnetospheric models to quantify model performance at the geosynchronous orbit with respect to observations of magnetic field strength and elevation. The models were compared to observations from two NOAA Geosynchronous Operational Environmental Satellites (GOES) in each event using the prediction efficiency and log-spectral distance metrics. The most recent GEM 2008–2009 challenge comparison [*Rastätter et al.*, 2013] quantified the ability of different physics-based and statistical models to reproduce the Dst geomagnetic activity index (http://wdc.kugi.kyoto-u.ac.jp/dstdir).

[5] Magnetospheric substorms are an essential part of the dynamics of near-Earth space [*McPherron*, 1991]: During the substorm growth phase, magnetic flux accumulates in the tail lobes due to dayside reconnection and is eventually released by rapid reconnection in the near-Earth magnetotail, which also starts the substorm expansion phase. The cross-tail current is diverted from the region of reconnection and flows along magnetic field lines to the midnight ionosphere where it flows westward for a (relatively) short distance enhancing locally the westward electroject before returning to the tail. Magnetospheric substorms and hence the dynamics of the near and far tail magnetosphere are also important from the point of view of space weather and can have a significant effect on technological systems even at ground level. For example, the largest geomagnetically induced currents (GIC) occur with highest probability during the substorm expansion phase about 5 min after the expansion onset below the corrected geomagnetic (CGM) latitude of 72° [*Viljanen et al.*, 2006].

[6] In this work, the performance of four global MHD models is systematically evaluated in the Earth's magnetosphere and in the ionosphere during an event with multiple substorms on 18 Feb 2004. The global MHD models BATS-R-US (Block Adaptive Tree Solar-wind Roe Upwind Scheme), GUMICS-4 (Grand Unified Magnetosphere-Ionosphere Coupling Simulation), LFM (Lyon-Fedder-Mobarry), and OpenGGCM (Open General Geospace Circulation Model) are given identical solar wind input, and the results are compared to the magnetic field measurements of Cluster 1 [*Balogh et al.*, 1997] within the magnetosheath, Geotail [*Kokubun et al.*, 1994] in the near tail, Wind [*Lepping et al.*, 1995] in the far tail, and the cross polar cap potential (CPCP) obtained from SuperDARN [*Chisham et al.*, 2007]. The models' magnetopause standoff distances are also compared to the empirical magnetopause model of *Lin et al.* [2010]. All simulations are carried out through NASA's Community Coordinated Modeling Center (CCMC) Run-on-Request system (http://ccmc.gsfc.nasa.gov), and the settings used for the models are as close to each other as reasonably possible. The results presented here are also available through CCMC. Similarly to *Pulkkinen et al.* [2011] and *Rastätter et al.* [2011], the detailed scientific analysis of the effect of various model parameters on the quality of model results in the magnetosphere is left for future work. As *Pulkkinen et al.* [2011] and *Rastätter et al.* [2011] studied the model performance at ground level and at geosynchronous orbit, this study is a natural next step to these investigations by validating the code performance in the near and far tail during dynamical events. In section 2, we describe the models, the features, and parameters which were used in this work and the event that was simulated. In section 4, we present the model results with the corresponding measurements, and in section 5, we compare them using the correlation coefficient (CC) and prediction efficiency (PE) metrics. We discuss the results and analysis in section 6 and draw our conclusions in section 7.

Table 1. Summary of Features and Settings of Global MHD Models Used in This Study, See the Text for Details

	BATS-R-US	GUMICS-4	LFM	OpenGGCM
MHD equations	ideal, conservative, $B_0 + B_1$	ideal, conservative, $B_0 + B_1$	ideal, semi-conservative, $B_0 + B_1$	semi-conservative with resistivity
Solver notes	eight-wave approximate Riemann	mostly Roe, subcycling, $\nabla \cdot B$ cleaning	total variation diminishing (TVD), constrained transport (CT)	TVD, CT
Order of MHD discretization: spatial / temporal	2 / 2	1 / 1	8 / 2	4 / 2
MHD grid	Cartesian, static, block-refined	Cartesian, dynamic, cell-refined	distorted spherical, static, not refined	stretched Cartesian, static, not refined
Dipole tilt updated with time	yes	no	yes[a]	no
Coordinate system of magnetosphere	GSM	GSE	SM	GSE

[a]The dipole orientation is fixed in SM coordinates, but solar wind and solar EUV conditions are adjusted with time.

2. Global MHD Model Features and Settings

[7] The features and settings of global MHD models used in this study are presented in Table 1. We emphasize that some of the models support a wide range of features and different settings, but we have listed only the ones used in this study. All models are executed through the CCMC Run-on-Request system and receive as input the solar wind data measured by the Advanced Composition Explorer (ACE) satellite [*Stone et al.*, 1998] located at GSE (221, –22, 9) R_E (Earth radii) during the simulated event provided by CCMC. The options and features used in all models are as close to each other as reasonably possible.

[8] While all the models solve the MHD equations in the magnetosphere and the same electrostatic potential equation in the ionosphere, there are some differences. Both BATS-R-US and GUMICS solve the ideal (i.e., inviscid and perfectly conducting), conservative, non-relativistic MHD equations [*Powell et al.*, 1999; *Janhunen et al.*, 2012], while LFM and OpenGGCM solve the MHD equations in a semi-conservative form where the total energy is replaced with the fluid energy [*Lyon et al.*, 2004; *Raeder et al.*, 2008]. In OpenGGCM, a resistive term is also included in the equation for the electric field [*Raeder et al.*, 2008]. BATS-R-US solves the MHD equations using a second-order eight-wave approximate Riemann solver that maintains zero divergence of the magnetic field to truncation error. GUMICS primarily uses a first-order seven-wave approximate Riemann solver and is the only model to periodically remove the divergence of magnetic field with the projection method of *Brackbill and Barnes* [1980]. Both LFM and OpenGGCM use a constrained transport method for advecting the magnetic field which preserves the divergence of magnetic field to roundoff error. BATS-R-US and GUMICS use an adapted Cartesian mesh for the magnetosphere. In BATS-R-US, the grid is adapted at the start of the simulation in blocks of 6^3 cells and is static afterwards, while in GUMICS, the grid is adapted during the simulation on a cell-by-cell basis based on local gradients of several plasma quantities and geometric considerations. LFM uses a distorted spherical grid and OpenGGCM uses a stretched Cartesian grid and neither uses AMR. BATS-R-US, GUMICS and LFM separate the magnetic field into perturbed and static background components [see *Tanaka* 1994]. The number of cells in the magnetospheric grid are about 800 k in BATS-R-US, 400 k in GUMICS, 330 k in LFM, and 3.6 M in OpenGGCM.

[9] The inner boundary of the magnetosphere in the models is between 2 and 4 R_E. The ionosphere and magnetosphere are coupled through field aligned currents (FAC) and electric potentials mapped between the ionosphere and the inner boundary of the magnetosphere along the Earth's dipole magnetic field. Field aligned currents are obtained from currents computed from the magnetic field in the inner magnetosphere. A two-dimensional electrostatic solver is used in the ionosphere to solve the ionospheric potential from FACs using the current continuity equation. The solved electric potential is used to set plasma flow in the inner magnetosphere. *Merkin and Lyon* [2010] provides more details on the LFM potential solver. OpenGGCM also includes a three-dimensional dynamical model of the thermosphere which adds, for example, the effect of the neutral wind dynamo to the ionospheric electric potential solution.

3. Event Description and Data

[10] Magnetospheric substorms are an essential part of the dynamics of near-Earth space [*McPherron*, 1991], and hence, an event with multiple substorms was selected to assess the performance of global models in the Earth's magnetosphere. The solar wind at the ACE satellite during the event of 18 Feb 2004 along with AE Dst indices is shown in Figure 1. The delay from ACE to the magnetopause for the event was calculated by *Honkonen et al.* [2011] to be 46 min. During the event, the solar wind density fluctuated between 1 and 3 cm^{-3} with large jumps

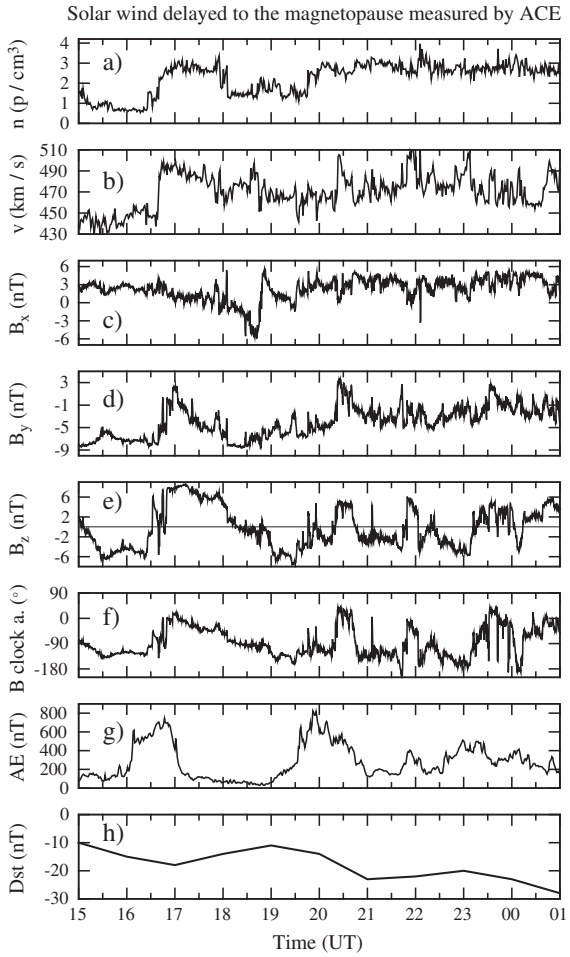

Figure 1. (a) Density, (b) velocity, (c) B_x, (d) B_y, (e) B_z, and (f) clock angle of the delayed (46 min) solar wind measured by ACE on 18 Feb 2004 at x_{GSE} = 221 R_E, (g) the provisional AE index and (h) final Dst index from Kyoto index service. Adapted from *Honkonen et al.* [2011].

recorded at 16:40 and 19:45 UT. The interplanetary magnetic field (IMF) z component changes sign more than a dozen times, varying between –8 and 8 nT. This leads to modest driving of the magnetosphere-ionosphere system as indicated by the AE index being over 600 for several hours during the event. Solar wind velocity is slightly above the average staying between 430 and 490 km/s.

[11] Undelayed solar wind is used as input for all models and, consequently, the delay from ACE to the solar wind boundary of the models must be taken into account separately. We assume a constant solar wind speed when calculating the delay for each model. In BATS-R-US the solar wind boundary is located at 33 R_E which translates to a delay of 2505 s. In GUMICS, LFM and OpenGGCM the solar wind boundaries are at 32, 30 and 60 R_E, which translate to delays of 2518, 2545 and 2145 s, respectively. These delays were used for all model results when comparing against observations.

[12] The simulation results reported here are available through the CCMC Web site as run numbers 102709_1, 103009_1, 011110_1, and 020410_1 prefixed with "Ilja_Honkonen_." The data from all simulations used in this study is saved to disk every 5 min, hence, all the satellite data is averaged using a 5 min sliding window. Figure 2 shows the trajectories of Cluster 1, Geotail and Wind during the simulated event with rectangles marking their location at the start of the event (15:00 UT). The distances of Cluster, Geotail, and Wind are about 9, 26, and 133 R_E from Earth, respectively. Cluster is flying towards dusk side ecliptic plane, Geotail is advancing outward from the dawn side flank, and Wind is almost stationary in the far tail.

4. Results

[13] Figure 3 shows the magnetic field components from simulations and Cluster 1 as a function of time, CC, and PE scores calculated from that data in section 5, scatter plots of each model B component versus the observation at the same instant of time, the coordinate of Cluster 1, and the region in which it is located. Based on Cluster 1 ion energy spectrogram data (not shown), Cluster was in the magnetosheath (labeled msheath in the Figure) with short excursions into the magnetosphere (labeled ms) until about 23:00 UT after which Cluster moved into the solar wind (labeled sw). Before 19:00 UT, there is a large difference between the models' B_x prediction at Cluster 1 but afterwards models and Cluster show a fairly constant B_x. B_x has two noticeable depressions at 20:30 and 22:00 UT of which the first is captured by BATS-R-US, GUMICS, and LFM and the second by OpenGGCM and perhaps by BATS-R-US. B_y shows several large changes between 17:00 and 23:00 UT. The largest change in B_y around 17:00 UT is captured by BATS-R-US, GUMICS, and LFM. The increase at 20:15 UT is reproduced best by BATS-R-US and GUMICS while in OpenGGCM and LFM, the increase

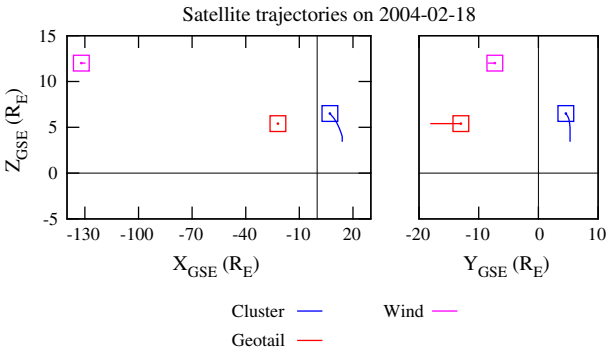

Figure 2. Trajectories of Cluster 1, Geotail, and Wind during the simulated event with rectangles marking their location at the start of the event (15:00 UT).

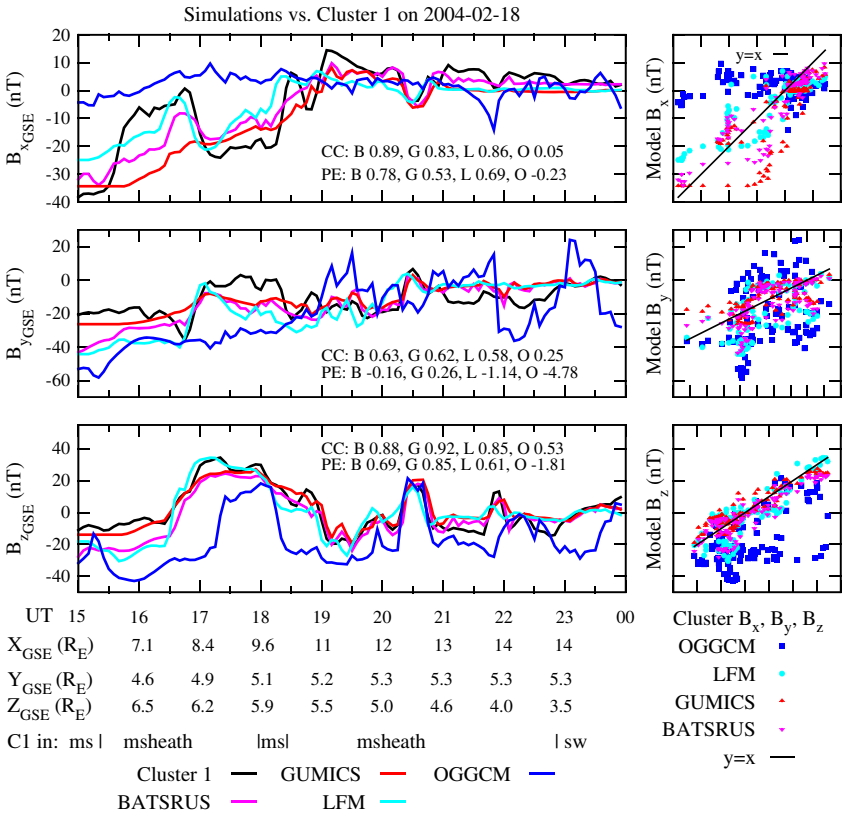

Figure 3. Five minute magnetic field data from Cluster 1 obtained by averaging spin average data over a 5 min window and magnetic field data at the same location from four global MHD simulations as function of time and scatter plots of each model B component versus the observation at the same instant of time. Correlation coefficients and prediction efficiencies between model results and observations are also shown, see section 5 for details. Location and region of Cluster 1 are shown at the bottom, see the text for an explanation of region codes.

starts some 20 min earlier. B_z shows three large increases and subsequent decreases starting around 16:30, 20:15, and 21:30 UT. The major features of B_z are captured by all models. After 17:00 UT, BATS-R-US and GUMICS are in good agreement with each other and along with LFM are close to Cluster 1 observations. OpenGGCM reproduces Cluster 1 observations reasonably well but with an offset of about –10 nT and a 1 to 2 h shorter first enhancement in B_z at 17:00 UT. At 20:30 UT, all models are in good agreement with Cluster 1. The temporary spread in modeled results at 18:30 UT stands out. For all components of B, the largest differences between models occur at the beginning of the event before about 17:00 UT.

[14] Figure 4 shows the magnetic field data from simulations and Geotail as a function of time. Based on density and ion temperature data, Geotail is mostly in the plasma sheet (labeled ps in the Figure) or plasma sheet boundary layer (labeled psbl) before 21:00 UT and in the lobe afterwards [*Aikio et al.*, 2013]. B_x has large changes around 16:00 and 20:00 UT of which only the changes before 17:30 UT seem to be reproduced by the models. All models predict the sharp decrease in B_x at Geotail around 17:00 UT. Geotail shows two depressions in B_x around 20:00 and 23:00 UT, but on the other hand, B_x stays more or less constant in BATS-R-US and LFM. Furthermore in GUMICS and OpenGGCM, contrary to Geotail, B_x clearly increases around 20:00 UT and also seems to increase, on average, around 23:00 UT. B_y does not show large features at Geotail except for noticeable increases at 16:00 and 17:00 UT. The former one seems to be captured by BATS-R-US and OpenGGCM while the latter is captured by LFM and, with a small delay, by BATS-R-US and GUMICS. The steady decrease of B_y between 17:00 and 19:30 UT is reproduced best by LFM with BATS-R-US and GUMICS also showing a similar feature. Geotail B_z has two large increases starting at 17:00 and 20:00 UT, which last for several hours. Between about 17:00 and 21:00 UT, all the models agree quite well with each other and, with an added offset of about 3 nT, also with Geotail. Interestingly, B_z in BATS-R-US, GUMICS, and LFM at Geotail is quite similar

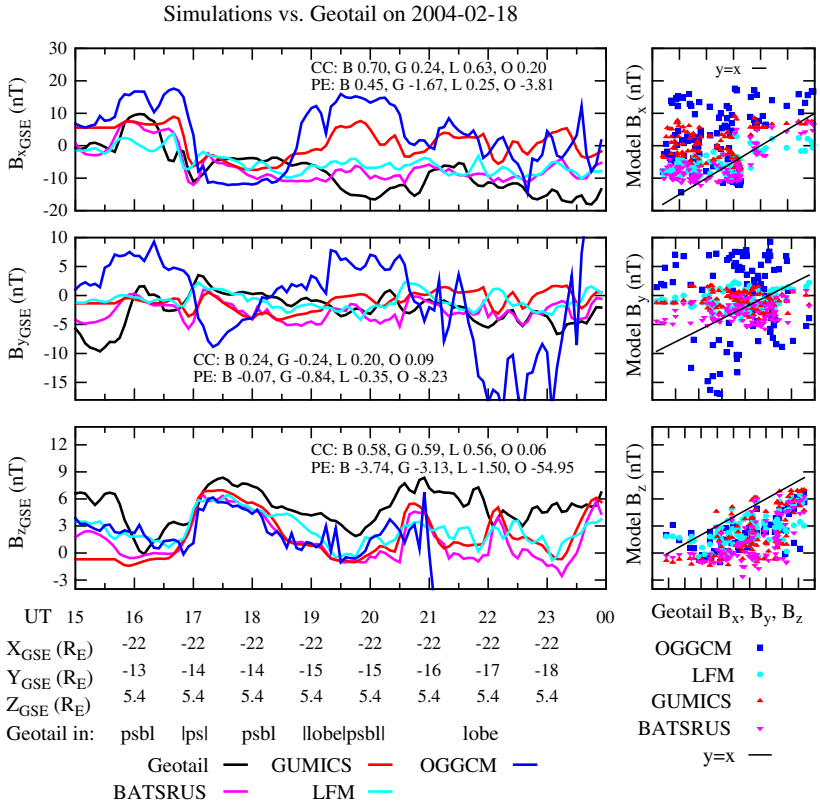

Figure 4. Magnetic field data from Geotail and simulations shown in the same format as in Figure 3.

to the simulated B_z at Cluster 1, but with a delay of 15 to 30 min. This is not the case for observations where Geotail shows one large enhancement between 20:00 and 22:00 UT while Cluster 1 shows two separate smaller enhancements at 20:30 and 22:00 UT. The disagreement between Geotail B_z and the models is largest before 16:00, at 21:30 and at 23:00 UT.

[15] Figure 5 shows the magnetic field data from simulations and Wind as a function of time. During the event, Wind travels between the northern or southern lobes via the neutral sheet (labeled nsheet in the Figure). On average B_x seems to decrease steadily between 16:00 and 19:00 UT after which it starts to increase. Several depressions of B_x are overlaid on top of the average behavior, the largest ones being at about 17:30, 19:30, 21:00, and 22:30 UT. The only features that seem to be reproduced consistently by all models are the depressions of B_x at 17:30 and 21:00 UT. OpenGGCM shows quite good agreement with Wind between 19:30 and 21:00 UT. Wind shows large increases in B_y starting at 17:00, 18:30, and 21:00 UT, and all models reproduce its observations between 17:00 and 18:30 UT. The second enhancement between 18:30 and 20:00 UT is not reproduced by any model. From about 20:00 onward BATS-R-US, GUMICS, and LFM again agree reasonably well with Wind. B_z measured by Wind has features similar to B_z of Cluster 1 but with a magnitude of about one fourth of that of Cluster. For the whole event, BATS-R-US, GUMICS, and LFM agree with Wind quite well with one exception of about 30 min around 20:00 UT. OpenGGCM agrees well with Wind between 17:00 and 19:30 UT after which it starts to show very large variations. Again the B_z result of BATS-R-US, GUMICS, and LFM are similar to their respective results for B_z at Cluster 1 but with a 30 to 45 min delay.

[16] Figure 6 shows the cross polar cap potential (CPCP) in the northern hemisphere from simulations and calculations from SuperDARN along with the number of flow vectors that were used in the calculation as a function of time. The procedure of calculating SuperDARN CPCP [*Ruohoniemi and Baker*, 1998] consist of finding the best fit for the electric potential Φ from observed flow vectors using $\vec{v} = -\frac{(\nabla\Phi)\times\vec{B}}{B^2}$. In regions without data coverage, a statistical model based on the solar wind IMF [*Ruohoniemi and Greenwald*, 1996] is used to constrain the solution. Most of the time, over 100 flow vectors are available and, for example, starting at 21:30 UT, about 300 vectors are available for almost 2 h. When the number of available vectors is above about 100, they seem to be available on both the dawn and dusk side of the northern hemisphere (data not shown).

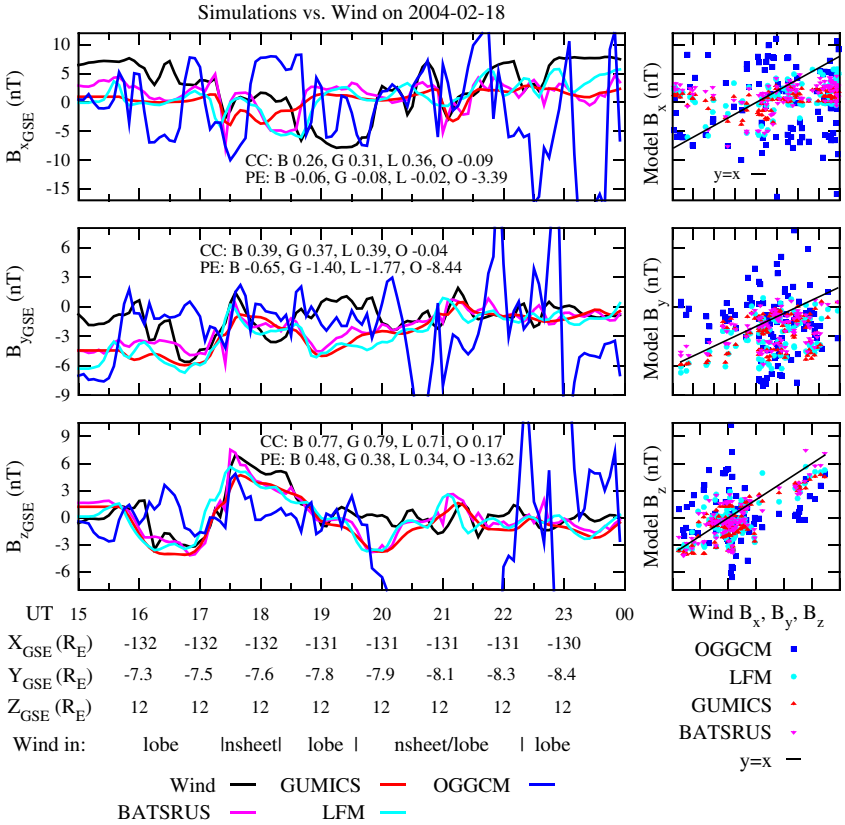

Figure 5. Magnetic field data from Wind and simulations shown in the same format as in Figure 3.

[17] During the event SuperDARN CPCP increases significantly four times with peak values at about 16:30, 19:30, 21:30, and 23:00 UT, which coincide with southward/northward turning of the IMF. A first-order minimum estimate for CPCP determined from the Defense Meteorological Satellite Program (DMSP) spacecraft available at NADIAWEB (http://cindispace.utdallas.edu/DMSP/NADIA_FAQ.html) gives values in the same range ($\sim$ 40 to $\sim$ 100 kV) as SuperDARN. BATS-R-US follows observations most closely with significantly different values for a period of about 30 min only at 17:30 and 20:45 UT. GUMICS and LFM capture the dynamics of CPCP well, but their results differ from observations by a constant factor of about 0.7 and 3, respectively. OpenGGCM also captures the main behavior of CPCP but with smaller enhancements before 21:00 UT and a very large increase in CPCP at around 23:00 UT.

[18] Figure 7 shows the minimum distance of the magnetopause from the Earth within 30° from the Sun-Earth line (referred to as R_0 from hereinafter) from simulations and the empirical model of *Lin et al.* [2010] as a function of time. With the exception of the first 1.5 h, BATS-R-US and GUMICS show very similar results for the entire event, staying between 11 and 13 R_E. Their result is also quite close to the empirical model, but with an almost constant offset of about 1.5 R_E. The results from LFM are closest to the empirical model with very good agreement (differences of less than 0.25 R_E) from about 17:00 to 20:30 UT and at other times the difference is at most about 1 R_E. In both, LFM and the empirical model, R_0 stays almost completely between 9 and 11 R_E. OpenGGCM has the lowest average value of R_0 of 8 R_E, and its dynamic range is much larger than the other models, varying between 6 and 12 R_E.

[19] The topology of the magnetic field in a global MHD simulation can give significant insight into the solution that was obtained and is essential for example when studying reconnection in a global setting [see e.g., *Dorelli et al.*, 2007]). Figures 8 and 9 show the magnetic field topology [*Rastätter et al.*, 2012] from simulations in the y = 0 R_E plane at about 21:20 UT (tracing parameters in Figure 8: N1 = N2 = 11, adaptation = 6; flow line start positions in Figure 9: uniform random in cut plane). In Figure 8, traced magnetic field lines connected to the Earth at both ends are shown in red, field lines connected only to the northern or southern hemisphere are shown in yellow or green, respectively, and field lines not connected to the Earth are shown in blue, while in Figure 9, field lines connected to either one hemisphere are shown in black. In GUMICS two large plasmoids form in the magnetotail starting at

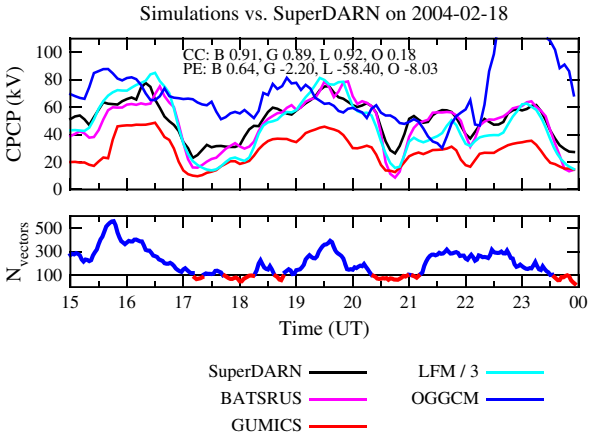

Figure 6. Five minute cross polar cap potential data from SuperDARN, obtained by averaging over a 5 min window, and CPCP data from four global MHD simulations as a function of time. Note that here LFM CPCP has been divided by 3. Correlation coefficients and prediction efficiencies between model results and observations are also shown, see section 5 for details. Also shown are the number of flow vectors that were used for calculating SuperDARN CPCP as a function of time.

20:30 and 23:30 UT, which is most likely caused by multiple large and fast rotations of the interplanetary magnetic field (IMF) clock angle [*Honkonen et al.*, 2011]. Figures 8b and 9b show the magnetic field topology from GUMICS at 21:12 UT, about 5 min before the first plasmoid dissipates. The plasmoid is visible as a region of closed magnetic field located in the far tail beyond –100 R_E. In BATS-R-US, the plasmoid forms about 10 min later than in GUMICS and is shown in Figures 8a and 9a 5 min before it dissipates. The result looks similar to that of GUMICS with a complicated structure of closed, lobe, and solar wind field lines in the ecliptic plane surrounded by lobe field lines above and below. Figures 8c and 9c show the plasmoid in LFM 5 min before it dissipates. The plasmoid forms at about the same time as that in BATS-R-US, but its structure is different. The closed field line regions stay closer to the ecliptic plane and do not detach from the Earth as in the case of BATS-R-US and GUMICS. At the time of plasmoid formation in other models, OpenGGCM does not show a closed field line region extending downstream from the Earth, but northern lobe field lines do show an additional region further down the tail. OpenGGCM shows significant changes in magnetic field topology prior to 20:00 UT. In BATS-R-US and LFM, the boundary between lobe and solar wind field lines in the magnetosheath is quite wavy, which is also noticeable in GUMICS.

5. Analysis

[20] In order to get a quantitative estimate for the performance of different models, the correlation coefficients (CC) and prediction efficiencies (PE) with respect to observations were calculated. Table 2 presents the correlation coefficients between model predictions and measurements. For every model, the magnetic field component with the largest correlation coefficient for every spacecraft is shown in bold and the smallest coefficient in italics. In this section we use the expressions best correlated and best predicted only for describing values of the respective metrics of one magnetic field component relative to another of the same combination of spacecraft and model. As will be shown, in some cases, an average prediction is better that the modeled result for all components of B, but even then, one component will have the highest score to which we will refer to as best.

[21] Overall B_z for every spacecraft is most often (7/12) best correlated with measurements, but there are spatial differences. On the dayside (Cluster 1), the largest correlations are evenly distributed between B_x and B_z, while B_y has the lowest correlation for three models. On the nightside close to the Earth (Geotail), B_x is most often (3/4) best correlated with measurements while B_y has most often (3/4) the worst correlation. Far in the magnetotail (Wind), B_z is best correlated with measurements and B_x the worst for all models. B_y is never the best correlated magnetic field component for any model and spacecraft combination.

[22] When examining the correlations of each magnetic field component separately, several things can be observed. For three of the four models the highest correlation in B_x is obtained on the dayside and the lowest correlation far in the tail. In BATS-R-US and LFM the correlation of B_x decreases steadily from dayside to far tail. In GUMICS the correlation of B_x drops significantly from dayside to nightside and increases slightly from nightside to the far tail, which is probably due to GUMICS having a smaller resolution at Geotail than the other models. In OpenGGCM the highest B_x correlation is on the

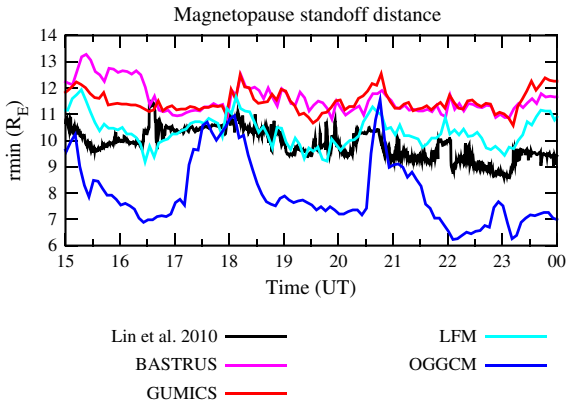

Figure 7. Minimum distance of the magnetopause from Earth within 30° from the Sun-Earth line in four global MHD simulations and the model of *Lin et al.* [2010] as a function of time.

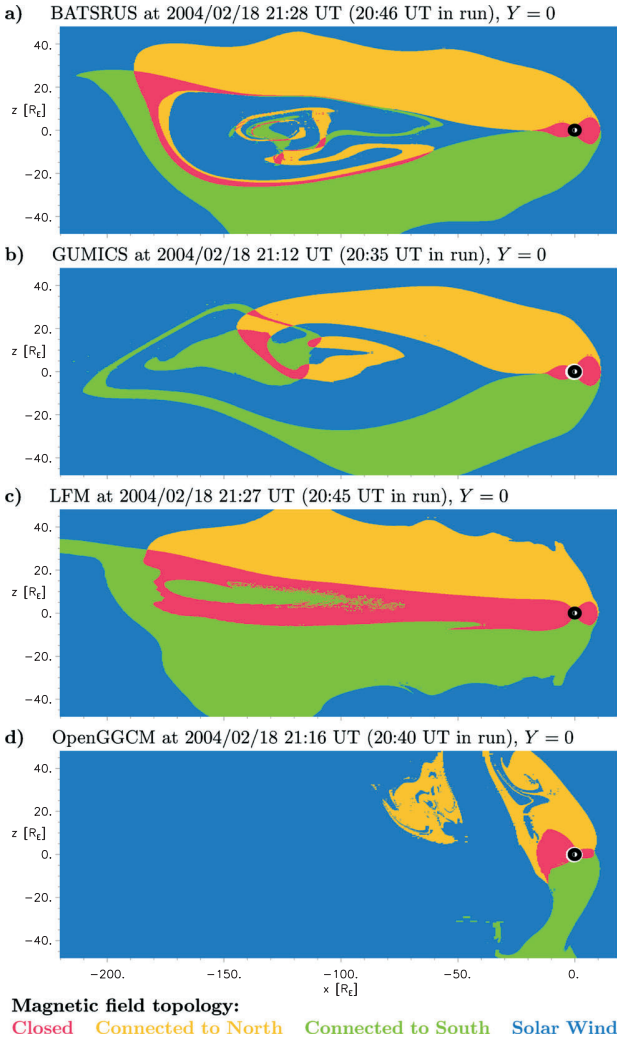

Figure 8. The color coded magnetic field topology from simulations in the $y = 0$ R_E plane. Traced magnetic field lines connected to the Earth at both ends are shown in red, field lines connected only to the northern (southern) hemisphere are shown in yellow (green), field lines not connected to the Earth are shown in blue. (a) BATS-R-US at 21:28 UT, (b) GUMICS at 21:12 UT, (c) LFM at 21:27 UT, and (d) OpenGGCM at 21:16 UT.

nightside. For all models, the lowest correlations of both B_y and B_z are on the nightside close to Earth with only one exception (OpenGGCM B_y at Wind). Overall, 37 out of 40 correlation coefficients are positive when SuperDARN is included.

[23] The prediction efficiency (PE) for a discrete signal is calculated following *Pulkkinen et al.* [2011]:

$$PE = 1 - \frac{< (x_{obs} - x_{sim})^2 >_i}{\sigma^2_{obs}}$$

where x_{obs} and x_{sim} are the observed and simulated signals, respectively, $< ... >_i$ indicates an arithmetic mean taken over i (i.e., time) and σ^2_{obs} is the variance of the observed signal. A PE value of 1 indicates a perfect prediction while a PE value of 0 is equal to using the mean value of the signal as a predictor. The prediction efficiencies of the models are presented in Table 3. For every model the magnetic field component with the largest prediction efficiency for every spacecraft is shown in bold and the smallest prediction efficiency in italics.

[24] Overall, B_x for every spacecraft is most often (7/12) best predicted by the models but again there are spatial differences. On the dayside and nightside close to the Earth, B_x is predicted best by three of the four models, but in the far tail, the highest prediction efficiency is mostly (3/4) obtained for B_z instead. On the dayside and far in the tail, B_y is predicted the worst almost without exception, but on the nightside close to Earth, B_z is predicted worst by all models. All the models are worse than using an average value for predicting B_y with the exception of one model for only one spacecraft (GUMICS on the dayside). On the other hand, for three models, the B_x and B_z prediction efficiencies on the dayside are above 0.5 and above 0.3 for B_z in the far tail.

[25] When examining the prediction efficiencies of each magnetic field component separately, several things are observed: For BATS-R-US and LFM, the highest PE for B_x is on the dayside and the lowest in the far tail, a situation identical to the correlation coefficients of B_x for these models. For GUMICS-4 and OpenGGCM, the highest PE for B_x is also on the dayside, but the lowest one is on the nightside close to Earth. The PE of B_y decreases downstream for GUMICS-4 and OpenGGCM but for BATS-R-US and LFM, B_y PE has the highest value on the nightside close to Earth. For all the models, B_z PE is significantly lower on the nightside than either the dayside or the far tail. Only BATS-R-US predicts the CPCP better than an average value would, and all other models are worse than using a random value. Overall, 13 out of 40 prediction efficiencies are positive when SuperDARN is included.

[26] For the combined magnetic field CC and PE results from all models B_x has the largest value among all components 12/24 times while B_y has the largest value 1/24 times and B_z 22/24 times. The number of times B_x, B_y and B_z have the smallest value among components are 5/24, 13/24 and 6/24 respectively.

[27] In order to estimate the effect that different modules/parameters of a model can have on the simulation result, three runs were done with a newer version of the BATS-R-US model (version 20110131) with different parameters: (1) only the version was changed, (2) higher resolution was used in the magnetosphere, and (3) the Rice Convection Model (RCM) module was included which solves the adiabatic drift of isotropic particle distributions [*Toffoletto et al.*, 2005] in the inner magnetosphere and provides density and pressure corrections to the magnetospheric module [*Tóth et al.*, 2005]. At Cluster 1, the standard resolution run has a resolution of 0.5 R_E (up

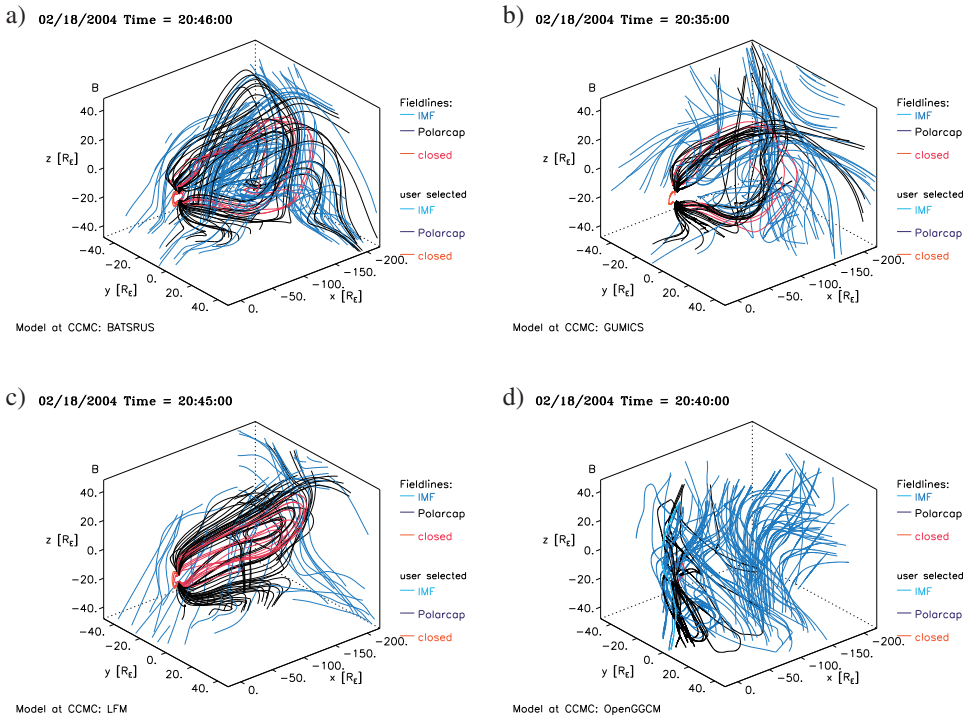

Figure 9. Magnetic field lines traced from the y = 0 R_E plane in simulations shown at same times as in Figure 8. Field lines connected to the Earth at both ends are shown in red, field lines connected only to the northern or southern hemisphere are shown in black, field lines not connected to the Earth are shown in blue.

to about 8 R_E distance from Earth) while the high resolution version has a resolution of 0.25 R_E (up to about 16 R_E distance from Earth on the dayside). At Geotail the resolutions are 1 R_E and 0.5 R_E, respectively although in the high resolution run the resolution decreases to 1 R_E when Z > 6 and further to 2 R_E when Z > 12 R_E.

[28] We only summarize the results for CC and PE in these runs, but the simulation results are again available through CCMC (run numbers 112112_1, 112112_2, 112112_3, and 112312_1 all prefixed with "Ilja_Honkonen_"). The results for B_x at Cluster orbit do not differ significantly between any version or module combination of BATS-R-US that was tested. At Geotail only RCM improved the results for B_x noticeably by increasing the absolute value of both CC and PE by about 0.1. For B_y, the differences are less straightforward: CC at Cluster improved slightly (about 0.05) with the new version regardless of the requested resolution and further by about 0.05 when using RCM. At Geotail, the new version improved CC of B_y by 0.15 which higher resolution

Table 2. Correlation Coefficients Between Model Magnetic Fields, Cross Polar Cap Potential, and Measurements for the Event of 18 Feb 2004[a]

		BATS-R-US	GUMICS-4	LFM	OpenGGCM
Cluster 1	B_x	**0.89**	0.83	**0.86**	*0.05*
	B_y	*0.63*	*0.62*	*0.58*	0.25
	B_z	0.88	**0.92**	0.85	**0.53**
Geotail	B_x	**0.70**	0.24	**0.63**	**0.20**
	B_y	*0.24*	*–0.24*	*0.20*	0.09
	B_z	0.58	**0.59**	0.56	*0.06*
Wind	B_x	*0.26*	*0.31*	*0.36*	*–0.09*
	B_y	0.39	0.37	0.39	–0.04
	B_z	**0.77**	**0.79**	**0.71**	**0.17**
SuperDARN	CPCP	0.91	0.89	0.92	0.18

[a]For every model, separately, the magnetic field component with the largest correlation coefficient for every spacecraft is shown in bold and the smallest coefficient in italics.

Table 3. Magnetic Field and Cross Polar Cap Potential Prediction Efficiencies of Global MHD Models for the Event of 18 Feb 2004[a]

		BATS-R-US	GUMICS-4	LFM	OpenGGCM
Cluster 1	B_x	**0.78**	0.53	**0.69**	**–0.23**
	B_y	*–0.16*	*0.26*	*–1.14*	*–4.78*
	B_z	0.69	**0.85**	0.61	–1.81
Geotail	B_x	**0.45**	–1.67	**0.25**	**–3.81**
	B_y	–0.07	**–0.84**	–0.35	–8.23
	B_z	*–3.74*	*–3.13*	*–1.50*	*–54.95*
Wind	B_x	–0.06	–0.08	–0.02	**–3.39**
	B_y	*–0.65*	*–1.40*	*–1.77*	–8.44
	B_z	**0.48**	**0.38**	**0.34**	*–13.62*
SuperDARN	CPCP	0.64	–2.20	–58.40	–8.03

[a]For every model, separately, the magnetic field component with the largest prediction efficiency for every spacecraft is shown in bold and the smallest efficiency in italics.

increased further by 0.06. Interestingly in this case, including RCM decreased CC by about 0.05. The prediction efficiency of B_y at Cluster improved significantly for the new version (by 0.35) and increasing the resolution and including RCM further improved the result by 0.1 and 0.2, respectively. At Geotail, the new version increased PE by almost 0.2 while increasing the resolution or including RCM decreased the result slightly. For B_z the results are relatively straightforward: A new version of BATS-R-US does not affect either CC or PE significantly; increasing the resolution improves all results noticeably at Cluster while having no or insignificant effect at Geotail. Interestingly, using RCM gives the worst results for B_z everywhere except for PE at Geotail, but there the results for all tested models are already worse than a random prediction.

6. Discussion

6.1. Metrics

[29] In this work the prediction efficiency (PE) and correlation coefficient (CC) metrics are used to obtain a quantitative estimate on model performance. An intuitive picture of these metrics can be obtained from the cross polar cap potential (CPCP) results where CC seems to quantify how well a model reproduces the "dynamics" of observations while PE indicates the quality of predicting the absolute values of observations. The CPCP prediction of LFM has the highest CC score of all models and indeed the relative changes of LFM CPCP correspond best to SuperDARN observations. On the other hand, the PE score of LFM CPCP is by far the lowest, which is not surprising given that LFM CPCP is mostly a factor of 3 larger than that of SuperDARN or any other model. As stated by *Pulkkinen et al.* [2011], no one metric is the absolute best and the choice of the metric depends on the situation. For some applications it could be a valid, albeit an unphysical, approach to divide the LFM CPCP by three in order to obtain the best available prediction for CPCP based on the CC metric.

[30] Overall, based on the PE metric, none of the tested models seem good at predicting observations on the nightside at a distance of about 25 R_E or more from the Earth. With the exception of B_x at Geotail and B_z at Wind, all model predictions at those satellites are worse than using an average value for the prediction, and even in the rest of the cases, PE is less than 0.5. On the dayside closer to Earth at about 14 R_E, all model PEs are significantly higher with the PE of B_x and B_z being over 0.5 for three out of four models. When the CC metric is used, all models fare substantially better, and interestingly, the highest and lowest values of CC occur in about the same locations and for the same components of B as with PE.

[31] In BATS-R-US and LFM the values of CC and PE tend to decrease steadily from the dayside to the far tail with the exception of B_z. For a system with high Mach number(s) flow past an obstacle, it is reasonable that model predictions are most accurate upstream of the obstacle and decrease downstream from there since turbulence and other effects have had more time to affect the system. In GUMICS and OpenGGCM, both CC and PE in the nightside close to the Earth often have smaller values than in the far tail. There does not seem to be a simple explanation for this behavior since GUMICS as a model is closer to BATS-R-US than LFM is to BATS-R-US (Table 1), but it is GUMICS that behaves differently from BATS-R-US in this respect.

[32] B_z is the largest exception to the above rule that metric scores decrease steadily downstream from the dayside since in all models and for both CC and PE, the score is higher in the far tail than in the nightside close to Earth. The lack of modeled physics in the nightside does not seem to explain this since including RCM in BATS-R-US decreases the nightside CC and PE scores if they are positive to begin with. One obvious explanation for this behavior could be the fact the axis of Earth's strong intrinsic dipole field is also directed in the general direction of the GSE Z axis. In order to verify this, a similar comparison would probably have to be carried out for Neptune's magnetosphere where the dipole axis can point almost directly sunward [*Ness et al.*, 1989].

6.2. Scatter Plots

[33] The scatter plots in Figures 3 to 5 illustrate the quality of predictions from another point of view. They allow one to understand the metric scores calculated here better and also let us estimate a limits to the CC and PE scores above which the simulation results can be considered good. For example, it is more apparent from the scatter plot than the time series that all models underestimate B_z at Geotail quite strongly. In a BATS-R-US run with RCM included, the shape of the point distribution does not change significantly but the whole distribution moves closer to the diagonal. Based on visual inspection of scatter plots, the modeled results seem good when both CC and PE are above about 0.6. Also using only the CC metric for estimating quality of the result can be misleading as the points in a scatter plot can still be quite far from the diagonal (e.g., GUMICS B_x at Cluster or BAST-R-US/LFM B_x at Geotail).

6.3. Cross Polar Cap Potential

[34] The CPCP result of GUMICS differs from that of SuperDARN by a constant factor of about 0.7 which can be explained, at least partially, by magnetospheric resolution and the dipole tilt angle. When using the Run-on-Request system, if the tilt angle is not updated with time, its direction is set to the start time of the simulation. In a previous simulation of this event with GUMICS, the dipole tilt angle was set to its average value during the simulated event [*Honkonen et al.*, 2011] and the CPCP prediction was noticeably higher, i.e., closer to SuperDARN and the other models. Also, increasing the resolution of the inner magnetosphere in GUMICS gives higher field-aligned currents (FAC), which increase CPCP further.

[35] The CPCP from SuperDARN reaches its saturation point of about 80 kV [e.g., *Shepherd et al.*, 2003] twice during the event at 16:30 and 19:30 UT. In this case the possible saturation of CPCP most likely would not change the relative result between models significantly because at those times all model predictions are less than or equal to SuperDARN with the exception of LFM, which is a factor of three higher. At times the number of flow vectors used for calculating SuperDARN CPCP falls below 100, and the number does not increase much beyond 300. This may cast some doubt on the calculated CPCP since, for example, the large statistical study of *Grocott et al.* [2012] only included periods with 300 or more flow vectors. We argue that the number of vectors and hence the reliability of SuperDARN CPCP is adequate for the purpose of comparing global MHD models, with quite different CPCP predictions, to observations.

6.4. Additional Physics

[36] *Pulkkinen et al.* [2011] reported that neither increasing spatial resolution in OpenGGCM nor including thermospheric physics in LFM systematically improved the performance of either model with respect to ground magnetic field observations. In this event increasing the resolution in BATS-R-US has a significant effect only on the dayside CC and PE of B_z, and in particular, higher resolution does not have a significant effect on B_x anywhere. Contrary to *Pulkkinen et al.* [2011], including the RCM module in BATS-R-US does not lead to an improved result in this case. Although the result for B_x does improve on the nightside by including RCM, the result for B_z becomes worse on the dayside. The reason for this behavior would be difficult to pinpoint based on even several tests. The possibilities range from small mistakes in the code, installation or usage to fundamental problems in the representation of the physics in each separate model, or in their coupling together. It is clear that including additional physical models in a simulation does not automatically guarantee a better result in the whole simulated volume.

6.5. Magnetopause Standoff Distance

[37] The magnetopause standoff distance between different MHD models varies by almost 6 R_E at 16:30 and 22:00 UT while at 18:00 UT all models show an almost identical value and are quite close at 20:30 UT. The standoff distance from an empirical model tends to fall in the middle of MHD models except from about 21:00 UT onward where the standoff distance is lower than in all but one MHD model. While proper validation would require a comparison to observations that is not possible in this case using the current CCMC interface. Nevertheless, a comparison to an empirical model shows what values and dynamical behavior to expect based on the upstream solar wind conditions.

[38] The increase in standoff distance of all models at 18:00 UT is probably due to the sudden large decrease in solar wind density while the increase at 20:30 UT might be due to northward turning of IMF B_z. The empirical model shows similar behavior although the increase at 18:00 UT is smaller than in MHD. At 22:00 UT the standoff distance again increases, probably due to northward turning of IMF B_z, in BATS-R-US, GUMICS, LFM, and the empirical model. While the standoff distance increases several times to a very similar value between all MHD models, the standoff distance seems to subsequently return to a baseline value that is different for each model. Finding the cause of this will require further investigation in subsequent works, as there is no apparent explanation for this behavior in the upstream solar wind conditions.

6.6. Statistical Studies Needed

[39] When validating, verifying, or just comparing models, as many parameters as possible should be kept constant. Unfortunately, this is difficult to accomplish with the models used here especially through the CCMC interface, which limits the parameter space of models available to users. For example, even when a higher resolution run of BATS-R-US is requested, the resolution does not increase in the whole simulation domain but is lower, for example, around the lobes. As shown in Table 1, there are also significant differences between, e.g., the magnetospheric grid used by different models. In this work we examined only event, and more may be needed in

order to draw solid conclusions on the performance of global MHD models in the Earth's magnetosphere. Due to the complicated physics involved and the differences in global models, it would be important to not only simulate single events but to run weeks or even many months worth of simulations in order to assess model performance using various metrics as a function of, for example, AE (http://wdc.kugi.kyoto-u.ac.jp/aedir) and Dst indices, the upstream solar wind driver, substorm phase, etc. This would allow the users of global models to estimate the quality of the solution for a particular event and could provide weights for statistical comparisons of simulations and observations. For model developers it would provide, for example, a baseline quality against which various modifications and changes in model parameters could be compared using a smaller but representative set of events.

[40] As a final note, there are four different global MHD models available at CCMC through a consistent interface, and each user can start several runs daily. Based on the results presented here and in previous works, there is virtually no reason to limit oneself to only one model when simulating the Earth's magnetosphere using CCMC resources. If two or more independent models agree on a particular result, it is very likely as close to reality as current state-of-the-art in global MHD can reasonably get.

6.7. Large Plasmoid Formation in Global MHD

[41] The results presented in Figures 8 and 9 lend more credibility to the hypothesis put forward by *Honkonen et al.* [2011] that multiple large and fast rotations of the IMF clock angle result in large plasmoid formation in a global MHD simulation. In three out of four different global MHD models, two large plasmoids form in the magnetotail during the event, and the plasmoids occur close in time between all three models in both cases although there are 5 to 15 min differences between the three models in the stages of plasmoid formation. The plasmoid structure is most similar between BATS-R-US and GUMICS with LFM also showing the closed magnetic field line region extending about –200 R_E downstream from Earth.

[42] The three models that agree on plasmoid formation also exhibit a large cross polar cap potential drop prior to the downstream growth of the closed magnetic field line region. GUMICS shows only very small variations in the ionospheric conductivities during the event, and hence changes to CPCP are almost completely due to changes in the FACs. Although the FACs in BATS-R-US vary by more than a factor of 2, the conductivities in the auroral oval also change moderately when the CPCP varies. In LFM both FACs and conductivities change significantly while in OpenGGCM only conductivities change drastically during variations of CPCP. Thus it seems that in order for a large tail plasmoid to be formed in a global MHD simulation, significant changes in ionospheric FACs are required and that ionospheric conductivities can have a significant effect on the structure of the formed plasmoid.

7. Conclusions

[43] In this work the performance of four global MHD models (BATS-R-US, GUMICS-4, LFM,and OpenGGCM) in the Earth's magnetosphere is studied by comparing model predictions to the magnetic field measurements of Cluster 1, Geotail, and Wind spacecraft during a multiple substorm event. Model results for the cross polar cap potential are also compared to the measurements of SuperDARN, and the model magnetopause standoff distances are compared to the empirical magnetopause model of *Lin et al.* [2010]. All simulations are executed through the CCMC Run-on-Request system. Comparisons are conducted using two quantitative and objective metrics: correlation coefficient and prediction efficiency. We find that for all four models, the best performance is on the dayside and, generally, model performance decreases steadily downstream from the Earth. From different components of the magnetic field, B_x is most often best predicted and correlated both on the dayside and the nightside close to the Earth whereas in all models B_z CC and PE are substantially higher in the far tail than for other components. On the dayside CCs are above 0.5 most of the time with B_x and B_z CCs close to 0.9 for three out of four models. In the magnetotail at a distance of about 130 R_E, the prediction efficiency of all models is below that of using an average value for the prediction with the exception of B_z. We also find that increasing the resolution or coupling an additional physical model does not automatically increase model performance at least with respect to the CC and PE metrics. With a coupled inner magnetosphere module, the performance of BATS-R-US increases significantly close to the Earth for B_y and in all relevant cases decreases moderately for B_z.

[44] **Acknowledgments.** This work is a part of the project 200141-QuESpace, funded by the European Research Council under the European Community's seventh framework programme. The work of I.H. and M.P. is supported by project 218165 of the Academy of Finland. A.G. is supported by NERC Grant NE/G019665/1. The National Center for Atmospheric Research is supported by the National Science Foundation. We thank the rest of the CCMC staff for providing a valuable service, the Cluster Active Archive, Coordinated Data Analysis Web and the instrument teams and PIs of Geotail MGF, Wind MFI, Cluster FGM, and ACE MAG and SWEPAM (S. Kokubun, R. Lepping, A. Balogh, N.F. Ness, D.J. McComas, respectively) for providing the spacecraft data used in this study. We also thank the World Data Center for Geomagnetism for the Kyoto AE and Dst index services and the anonymous referee for insightful comments. I.H. thanks C. Anekallu for insightful discussions.

References

Aikio, A. T., T. Pitkänen, I. Honkonen, M. Palmroth, and O. Amm (2013), IMF effect on the polar cap contraction and expansion during a period of substorms, *Ann. Geophys.*, submitted.

Balogh, A., et al. (1997), The cluster magnetic field investigation, *Space Sci. Rev.*, *79*, 65–91, doi:10.1023/A:1004970907748.

Brackbill, J. U., and D. C. Barnes (1980), The effect of nonzero product of magnetic gradient and B on the numerical solution of the magnetohydrodynamic equations, *J. Comput. Phys.*, *35*, 426–430.

Chisham, G., et al. (2007), A decade of the Super Dual Auroral Radar Network (SuperDARN): Scientific achievements, new techniques

and future directions, *Surv. Geophys.*, *28*(1), 33–109, doi:10.1007/s10712-007-9017-8.

Dorelli, J. C., A. Bhattacharjee, and J. Raeder (2007), Separator reconnection at Earth's dayside magnetopause under generic northward interplanetary magnetic field conditions, *J. Geophys. Res.*, *112*, A02202, doi:10.1029/2006JA011877.

Grocott, A., S. E. Milan, S. M. Imber, M. Lester, and T. K. Yeoman (2012), A quantitative deconstruction of the morphology of high-latitude ionospheric convection, *J. Geophys. Res.*, *117*, A05317, doi:10.1029/2012JA017580.

Honkonen, I., S. von Alfthan, A. Sandroos, P. Janhunen, and M. Palmroth (2013), Parallel grid library for rapid and flexible simulation development, *Comput. Phys. Commun.*, *184*, 1297–1309, doi:10.1016/j.cpc.2012.12.017.

Honkonen, I., M. Palmroth, T. I. Pulkkinen, P. Janhunen, and A. Aikio (2011), On large plasmoid formation in a global magnetohydrodynamic simulation, *Ann. Geophys.*, *29*(1), 167–179, doi:10.5194/angeo-29-167-2011.

Janhunen, P., M. Palmroth, T. Laitinen, I. Honkonen, L. Juusola, G. Facskó, and T. I. Pulkkinen (2012), The GUMICS-4 global MHD magnetosphere-ionosphere coupling simulation, *J. Atmos. Sol. Terr. Phys.*, *80*, 48–59, doi:10.1016/j.jastp.2012.03.006.

Kokubun, S., T. Yamamoto, M. H. Acuña, K. Hayashi, K. Shiokawa, and H. Kawano (1994), The GEOTAIL magnetic field experiment, *J. Geomag. Geoelectr.*, *46*, 7–21.

Krausmann, E., and V. Bothmer (2012), Meeting report: European Commission's space-weather awareness dialogue, *Space Weather*, *10*, S04006, doi:10.1029/2012SW000790.

Ledvina, S. A., Y.-J. Ma, and E. Kallio (2008), Modeling and simulating flowing plasmas and related phenomena, *Space Sci. Rev.*, *139*(1-4), 143–189, doi:10.1007/s11214-008-9384-6.

Lepping, R. P., et al. (1995), The WIND magnetic field investigation, *Space Sci. Rev.*, *71*, 207–229, doi:10.1007/BF00751330.

Lin, R. L., X. X. Zhang, S. Q. Liu, Y. L. Wang, and J. C. Gong (2010), A three-dimensional asymmetric magnetopause model, *J. Geophys. Res.*, *115*, A04207, doi:10.1029/2009JA014235.

Lyon, J. G., J. A. Fedder, and C. M. Mobarry (2004), The Lyon-Fedder-Mobarry (LFM) global MHD magnetospheric simulation code, *J. Atmos. Sol. Terr. Phys.*, *66*(15-16), 1333–1350, doi:10.1016/j.jastp.2004.03.020.

McPherron, R. L. (1991), *Physical Processes Producing Magnetospheric Substorms and Magnetic Storms*, Geomagnetism, vol. 4, J. Jacobs, Academic Press Ltd., London, England.

Merkin, V. G., and J. G. Lyon (2010), Effects of the low-latitude ionospheric boundary condition on the global magnetosphere, *J. Geophys. Res.*, *115*, A10202, doi:10.1029/2010JA015461.

Ness, N. F., M. H. Acuña, L. F. Burlaga, J. E. P. Connerney, R. P. Lepping, and F. M. Neubauer (1989), Magnetic fields at Neptune, *Science*, *246*, 1473–1478, doi:10.1126/science.246.4936.1473.

Powell, K. G., P. L. Roe, T. J. Linde, T. I. Gombosi, and D. L. De Zeeuw (1999), A solution-adaptive upwind scheme for ideal magnetohydrodynamics, *J. Comput. Phys.*, *154*, 284–309, doi:10.1006/jcph.1999.6299.

Pulkkinen, A., et al. (2011), Geospace Environment Modeling 2008-2009 Challenge: Ground magnetic field perturbations, *Space Weather*, *9*, S02004, doi:10.1029/2010SW000600.

Raeder, J., D. Larson, W. Li, E. L. Kepko, and T. Fuller-Rowell (2008), OpenGGCM simulations for the THEMIS mission, *Space Sci. Rev.*, *141*(1-4), 535–555, doi:10.1007/s11214-008-9421-5.

Rastätter, L., et al. (2013), Geospace environment modeling 2008–2009 challenge: Dst index, *Space Weather,* accepted, *11*(4), 187–205, doi:10.1002/swe.20036.

Rastätter, L., M. M. Kuznetsova, D. G. Sibeck, and D. H. Berrios (2012), Scientific visualization to study flux transfer events at the Community Coordinated Modeling Center, *Adv. Space Res.*, *49*(11), 1623–1632, doi:10.1016/j.asr.2011.12.034.

Rastätter, L., M. M. Kuznetsova, A. Vapirev, A. Ridley, M. Wiltberger, A. Pulkkinen, M. Hesse, and H. J. Singer (2011), Geospace environment modeling 2008–2009 challenge: Geosynchronous magnetic field, *Space Weather*, *9*, S04005, doi:10.1029/2010SW000617.

Ruohoniemi, J. M., and K. B. Baker (1998), Large-scale imaging of high-latitude convection with Super Dual Auroral Radar Network HF radar observations, *J. Geophys. Res.*, *103*, 20797–20811, doi:10.1029/98JA01288.

Ruohoniemi, J. M., and R. A. Greenwald (1996), Statistical patterns of high-latitude convection obtained from Goose Bay HF radar observations, *J. Geophys. Res.*, *101*, 21743–21763, doi:10.1029/96JA01584.

Ryu, D., and T. W. Jones (1995), Numerical magetohydrodynamics in astrophysics: Algorithm and tests for one-dimensional flow, *The Astrophys. J.*, *442*, 228–258, doi:10.1086/175437.

Shepherd, S. G., J. M. Ruohoniemi, and R. A. Greenwald (2003), Testing the Hill model of transpolar potential with Super Dual Auroral Radar Network observations, *Geophys. Res. Lett.*, *30*(1), 1002, doi:10.1029/2002GL015426.

Stone, E. C., A. M. Frandsen, R. A. Mewaldt, E. R. Christian, D. Margolies, J. F. Ormes, and F. Snow (1998), The advanced composition explorer, *Space Sci. Rev.*, *86*, 1–22, doi:10.1023/A:1005082526237.

Tanaka, T. (1994), Finite volume TVD scheme on an unstructured grid system for three-dimensional MHD simulation of inhomogeneous systems including strong background potential fields, *J. Comput. Phys.*, *111*, 381–389.

Toffoletto, F., S. Sazykin, R. Spiro, and R. Wolf (2005), Inner magnetospheric modeling with the Rice Convection Model, *Space Sci. Rev.*, *107*(1-2), 175–196, doi:10.1023/A:1025532008047.

Tóth, G., et al. (2005), Space weather modeling framework: A new tool for the space science community, *J. Geophys. Res.*, *110*, A12226, doi:10.1029/2005JA011126.

Viljanen, A., E. I. Tanskanen, and A. Pulkkinen (2006), Relation between substorm characteristics and rapid temporal variations of the ground magnetic field, *Ann. Geophys.*, *24*(2), 725–733, doi:10.5194/angeo-24-725-2006.

Welling, D. T., J. Koller, and E. Camporeale (2012), Verification of Spacepy's radial diffusion radiation belt model, *Geosci. Model Dev.*, *5*(2), 277–287, doi:10.5194/gmd-5-277-2012.

Article IV

On large plasmoid formation in a global magnetohydrodynamic simulation,
Annales Geophysicae, volume 29, issue 1, 2011, pages 167-179.

Ann. Geophys., 29, 167–179, 2011
www.ann-geophys.net/29/167/2011/
doi:10.5194/angeo-29-167-2011

Annales Geophysicae

On large plasmoid formation in a global magnetohydrodynamic simulation

I. Honkonen[1,2], **M. Palmroth**[1], **T. I. Pulkkinen**[1,*], **P. Janhunen**[1], **and A. Aikio**[3]

[1]Finnish Meteorological Institute, Helsinki, Finland
[2]Department of Physics, University of Helsinki, Helsinki, Finland
[3]Department of Physical Sciences, University of Oulu, Oulu, Finland
[*]currently at: School of Electrical Engineering, Aalto University, Finland

Received: 25 May 2010 – Revised: 9 December 2010 – Accepted: 10 January 2011 – Published: 14 January 2011

Abstract. We investigate plasmoid formation in the magnetotail using the global magnetohydrodynamic (MHD) simulation GUMICS-4. Here a plasmoid implies a major reconfiguration of the magnetotail where a part of the tail plasma sheet is ejected downstream, in contrast to small Earthward-propagating plasmoids. We define a plasmoid based solely on the structure of the closed (connected to the Earth at both ends) magnetic field line region. In this definition a plasmoid is partly separated from the ordinary closed field line region by lobe field lines or interplanetary field lines resulting from lobe reconnection. We simulate an event that occurred on 18 February 2004 during which four intensifications of the auroral electroject (AE) index occurred in 8 h. Plasmoids form in the simulation for two of the four AE intensifications. Each plasmoid forms as a result of two consecutive large and fast rotations of the interplanetary magnetic field (IMF). In both cases the IMF rotates 180 degrees at 10 degrees per minute, first from southward to northward and some 15 min later from northward to southward. The other two AE intencifications however are not associated with a plasmoid formation. A plasmoid does not form if either the IMF rotation speed or the angular change of the rotation are small. We also present an operational definition for these fully connected plasmoids that enables their automatic detection in simulations. Finally, we show mappings of the plasmoid footpoints in the ionosphere, where they perturb the polar cap boundary in both hemispheres.

Keywords. Magnetospheric physics (Magnetospheric configuration and dynamics; Magnetotail)

Correspondence to: I. Honkonen
(ilja.honkonen@fmi.fi)

1 Introduction

The solar wind energy drives all dynamic phenomena within the near-Earth space. The basic process of extracting solar wind energy is called a substorm, during which solar wind energy is first loaded into the magnetosphere during the growth phase and subsequently released during the substorm expansion phase (McPherron, 1991). During the recovery phase, the magnetospheric and ionospheric dynamics subside. The growth phase starts typically after the interplanetary magnetic field (IMF) turns southward, and the dayside reconnection begins to bring more magnetic energy to the tail lobes. The new magnetic flux added to the tail lobes stretches the tail and compresses the plasma sheet, increasing the intensity of the duskward cross-tail current as well. At some point, the tail current disrupts, leading to two phenomena (McPherron, 1991): (1) the cross-tail current is forced to divert via the ionosphere and (2) a part of the tail is released downwind. These ejected large magnetic structures are called plasmoids. The onset mechanism of the current disruption and the chain of events prior to the onset are still unknown and under vigorous research. While Hsu and McPherron (2002) showed that half of all substorms are triggered by northward turning of the interplanetary magnetic field (IMF), half of the substorms are associated with no particular changes within the solar wind driver and may be driven by an internal plasma instability within the magnetotail (e.g. Coppi et al., 1966).

Plasmoids are large magnetic structures that form in the Earth's magnetotail and remove plasma and energy from the magnetosphere (Hones, 1979). During a plasmoid formation, the three-dimensional structure of the magnetotail becomes complicated, with spatially alternating closed and open magnetic topologies. Figure 1 shows the traditional two-dimensional description of a plasmoid as closed loops

Published by Copernicus Publications on behalf of the European Geosciences Union.

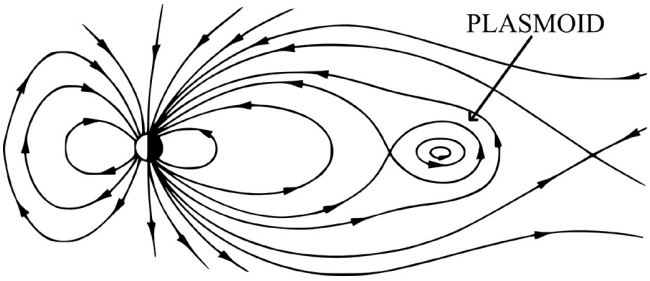

Fig. 1. A plasmoid forming inside the closed magnetic field line region in the y-plane. Original figure by D. P. Stern.

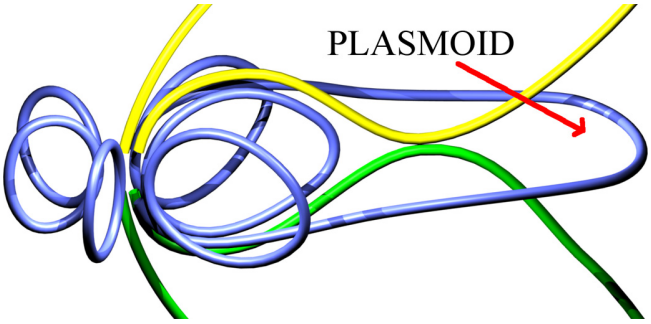

Fig. 2. A plasmoid consisting of the closed magnetic field line region partially detached from the Earth. Closed magnetic field lines are colored blue. Magnetic field lines attached to the Earth only in the parallel and anti-parallel directions (lobe field lines) are colored yellow and green, respectively.

of magnetic field lines that are not connected to the Earth (original figure by D. P. Stern). The closed loops form when regions of the plasma sheet are severed from the Earth by magnetic reconnection in the near-Earth tail (Hones et al., 1984a). In three dimensions, with non-zero B_y-component in the plasma sheet, reconnection creates a flux rope of closed field lines that remain connected to the Earth (Hughes and Sibeck, 1987). Each magnetic field line belongs to one of the following topologies: (1) closed field line with both ends in the ionosphere, (2) IMF field line with both ends in the solar wind, (3)–(4) lobe field line with one end in the ionosphere and the other in the solar wind, (5) a closed loop that is not attached to the Earth. If the symmetry ($B_y = 0$) that leads to topology (5) is removed, plasmoid field lines belong to topologies 1...4, which often seems to be the case (Moldwin and Hughes, 1992). Figure 2 shows a sketch of this situation. Therefore, without $B_y = 0$ symmetry plasmoids in the general 3-D case cannot be defined by the type of the magnetic field lines alone. A plasmoid could be defined, for example, as magnetic field lines that cross the equatorial plane more than once. However, this definition is arbitrary as it depends on the chosen plane and the number of crossings (Birn et al., 1989).

The size of plasmoids has varied greatly in observations, e.g. in the x-direction from 4...10 R_E (Ieda et al., 1998) and $16.7 \pm 13.0\,R_E$ (Moldwin and Hughes, 1992) up to 75...150 R_E (Hones et al., 1984b), depending on their distance from the Earth. Estimations of the energy carried away by plasmoids have also varied by an order of magnitude, from about 0.2×10^{15} J (Ieda et al., 1998) to 4×10^{15} J (Silbergleit et al., 1997). In these studies plasmoids were detected mainly from the magnetic field and plasma data of ISEE-3 or Geotail spacecraft. The large error estimates were due to the fact that data was only available from a single spacecraft.

Although it is well established that plasmoids are mostly associated with substorms (Slavin et al., 1987; Moldwin and Hughes, 1993), typically it is only stated that plasmoids form due to reconnection of closed magnetic field lines in the near-Earth tail (Hones, 1977, 1979; Hones et al., 1984b; Birn et al., 1989; Moldwin and Hughes, 1992; Slinker et al., 1995; Ieda at al., 2001; Farr et al., 2008). In this paper we examine the details of plasmoid formation using simulation results for the event of 18 February 2004, from the global magnetohydrodynamic simulator GUMICS-4 (Janhunen, 1996). We define a plasmoid as a major reconfiguration of the magnetotail where part of the tail plasma sheet is ejected downstream, in contrast to Earthward-propagating small plasmoids (see for example Zong et al., 2004). First we describe the event of 18 February 2004, the delay from ACE to the magnetopause and compare simulation results with Cluster observations. Then we present a method for identifying tail plasmoids in simulations based on the structure of the closed magnetic field lines, e.g. connected plasmoids, and show the large-scale evolution of the magnetotail in the simulation. Finally, we show the effects of the plasmoid footpoints in the simulation ionosphere and discuss plasmoid formation in light of observations.

2 Event description and the solar wind driver

Figure 3 illustrates the calculation of the delay from ACE spacecraft (located at GSE $(221, -22, 9)\,R_E$) to the magnetopause. The delay is computed by correlating upstream ACE measurements with those of GOES-12 and Cluster 1. Upstream measurements are delayed 0...4000 s in 10 s increments and a linear least squeares value is calculated for all delays. GOES-12 was near local noon at the time of arrival of a solar wind pressure pulse. The pulse, detected at Earth at 16:40 UT, caused an increase of the dayside magnetic field in the magnetosphere (Fig. 3a). The linear least squares value is minimized with a delay of 2720 s, about 45 min.

Cluster 1 traversed the dayside magnetosheath during the period 19:30 to 22:30 UT. During that period Cluster 1 observed several sign changes in B_z, which are correlated to

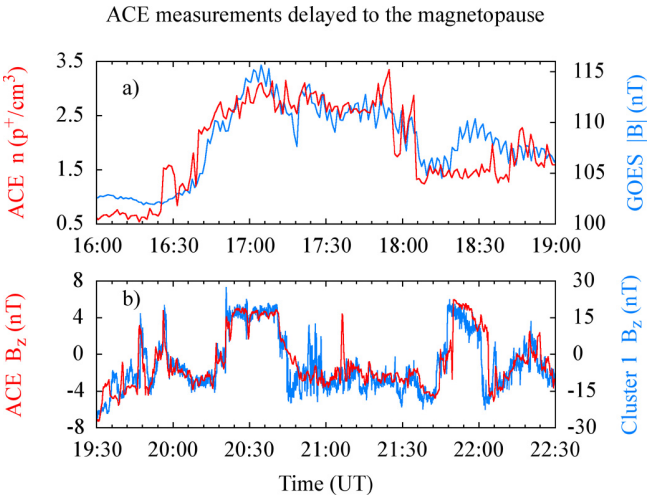

Fig. 3. The solar wind n and B_z from ACE delayed by about 45 and 46 min over n from GOES-12 and B_z from Cluster 1, respectively. Note that the panels are displayed at different times.

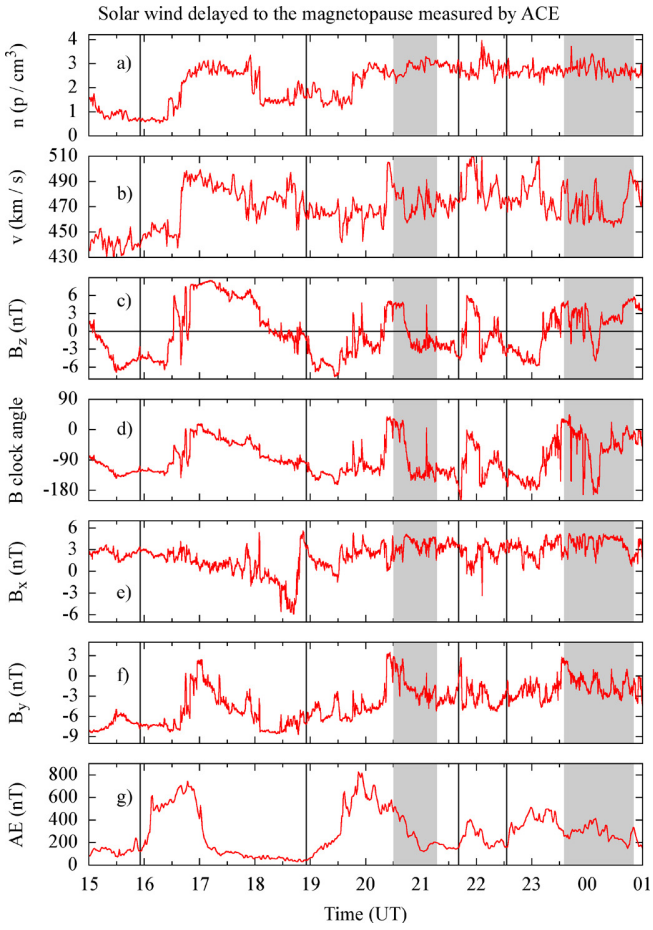

Fig. 4. **(a)** density, **(b)** velocity, **(c)** B_z, **(d)** clock angle, **(e)** B_x, **(f)** B_y of the delayed solar wind measured by ACE on 18 February 2004 at $x_{GSE} = 221\ R_E$ and **(g)** the provisional AE index from Kyoto AE index service. Vertical black lines mark the start of the AE intensifications, vertical grey bars mark the plasmoids formed in the simulation.

ACE measurements. The linear least squares value between ACE and Cluster 1 B_z is minimized using a delay of 2770 s, about 46 min (Fig. 3b). The average of the two estimates (2745 s or 46 min) is used as the delay to the magnetopause in the following analysis. Figure 3b shows that the delayed ACE B_z measurements precede those of Cluster 1 before 20:00 UT and lag behind Cluster 1 measurements after 21:30 UT. This implies an increase in the solar wind bulk speed during that time, which indeed is the general trend (see Fig. 4).

Figure 4 shows the solar wind data from ACE delayed to the magnetopause using the 46 min delay. The solar wind density (Fig. 4a) fluctuated between 1 and 3 cm^{-3} during the event with large jumps recorded at 16:40 and 19:45 UT. The solar wind bulk speed (Fig. 4b) increased from 440 km s^{-1} to 500 km s^{-1} at 16:40 UT and fluctuated between 460 and 500 km s^{-1} for the rest of the day. The most striking features in the solar wind parameters during this event were changes in the IMF. Its z-component (Fig. 4c) changed sign more than a dozen times, varying between -8 and 8 nT. The IMF clock angle (Fig. 4d) varied between -140 and 20° several times during the event. The clock angle is defined by the direction of the IMF in the GSE yz-plane as 0° in the positive z-direction and ranging from $-180°$ to 180°. The x-component of IMF did not show large features except for a sinusoidal change around 18:30 UT.

The provisional AE index on 18 February 2004 presented in Fig. 4g shows four intensifications between 16:00 and 24:00 UT. The first AE intensification at 16:00 UT does not seem to be directly driven by the solar wind, as all parameters are fairly constant during that time. The increse (growth phase) of the second AE intensification starts as the IMF turns southward at 18:50. This is followed by the sudden intensification of AE (substorm expansion phase) about 45 min later when B_z suddenly jumps from -7 to -3 nT. The recovery phase starts around 20:30 after the IMF has turned northward.

3 Simulation

3.1 GUMICS-4

The GUMICS-4 global magnetosphere-ionosphere coupling simulation (Janhunen, 1996, and references therein) solves the ideal MHD equations in fully conservative form. The simulation is robust and has been tested extensively against observations (e.g. Palmroth et al., 2003). Elliptic cleaning is used to enforce $\nabla \cdot \boldsymbol{B} = 0$ (Brackbill and Barnes, 1980). GUMICS-4 uses a hierarchical cubic grid and also temporal subcycling which reduces the required computation time

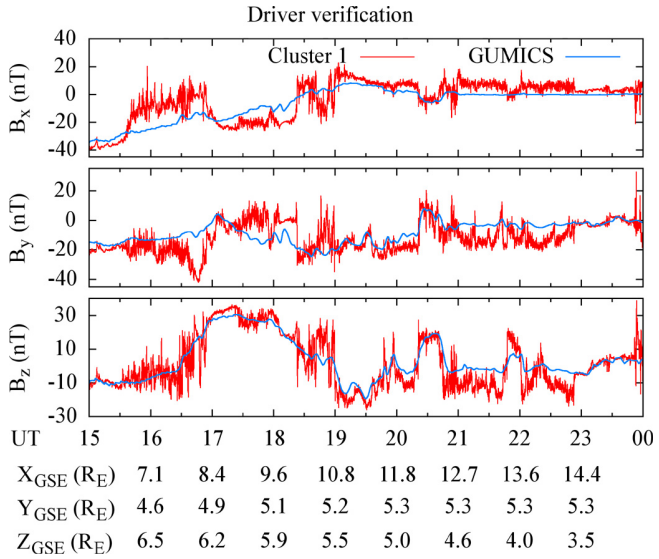

Fig. 5. The magnetic field from Cluster 1 and the simulation at the same location as function of time.

by at least an order of magnitude when compared to global timestepping. The simulation grid extends from 32 R_E upstream to 224 R_E downstream from the Earth and is 128 R_E in the y_{GSE} and z_{GSE} directions. The simulation grid uses automatic cell refinement and the grid is adapted at run time based on local gradients, spatial coordinates and user-specified priority functions (Janhunen et al., 1996). The finest spatial resolution in the simulation used in this study is 1/4 R_E. GUMICS-4 takes as input the solar wind density, temperature, velocity and magnetic field at the upstream boundary and outputs the plasma parameters in the simulation box.

The inner boundary of the MHD simulation at 3.7 R_E is coupled to the ionospheric solver. The electrostatic ionosphere model is coupled to the magnetosphere through field-aligned currents and electron precipitation. The ionospheric electron density is solved in a 3-D grid with 20 non-uniform altitude levels, taking into account the magnetospheric field-aligned current, source plasma density and temperature (Janhunen, 1996). Height-integrated Hall and Pedersen conductivities are obtained from the electron density and are used along with the field aligned current to solve the horizontal current distribution in the ionosphere. This gives the ionospheric potential which is used as the electric field in the MHD equations.

The GUMICS simulation was run using the solar wind observations of ACE from 14:00 to 24:00 UT on 18 February 2004. The IMF x-component was set to a constant value of 0 nT in order to keep the magnetic field at the upstream boundary of the simulation divergence free. The dipole tilt angle was set at the start of the simulation and kept at a value of $-4.3°$ for the entire run.

3.2 Driver verification

Figure 5 shows the simulation results along Cluster 1 orbit together with the measurements during the event. The observed B_z component measured by the FGM instrument (Balogh et al., 2001) is well reproduced by GUMICS, as are some of the dynamics of the B_y component. The general trend in B_x is also reproduced by GUMICS. The observations of Geotail at (-22.2...-21.4, -10.8...-18.2, 5.3...5.3) R_E and Wind at (-132.6...-130.0, -6.8...-8.6, 11.6...11.7) R_E (both in GSE) are also reproduced by GUMICS quite well (Palmroth et al., 2010). On the whole the effects of the solar wind driver are reproduced in the simulation, giving credibility to the modelling results presented in this paper. However the key point here is to demonstrate that the solar wind driver measured at L1 far upstream of the Earth is indeed the same that impacted the magnetosphere 45 min later.

4 Plasmoids in global MHD

4.1 Definition

Plasmoid development can be divided into three distinct stages (Hughes and Sibeck, 1987): (1) reconnection within the closed magnetic field line region in the plasma sheet with $B_y \neq 0$ creates helical field lines that are connected to the Earth along the flanks. This structure is known as a flux rope. A plasmoid is formed at this stage if it is defined, for example, as field lines that cross the equatorial plane more than once. (2) As the reconnection in (1) proceeds to lobe field lines the flux rope becomes enveloped in IMF field lines and also lobe field lines (Birn et al., 1989) in the central tail, and the flux rope starts to move tailward. At this stage the plasmoid is still connected to the Earth at both ends and can be identified from the topology of closed field lines alone, using a method which we will present next. The plasmoid is formed at this stage at the latest. (3) At some point the plasmoid will start to dissipate due to reconnection near the flanks between the plasmoid closed field lines and lobe field lines. Using the definition we present below, the plasmoid no longer exists at this stage (it is not connected to the Earth at both ends), but its remnants on IMF and lobe field lines can still be identified (Birn et al. 1989; Farr et al. 2008). In this work we will refer to flux ropes in stage 2), e.g. connected to the Earth at both ends, as plasmoids.

We define a plasmoid using the magnetic field topology, as a region of closed field lines detached from the quasidipolar tail region. Normally, closed magnetic field lines are bounded by a surface which encloses the Earth and the inner magnetosphere along quasi-dipolar field lines. Topologically, the surface has genus zero (e.g. a sphere). Reconnection within the plasma sheet breaks the surface, creating a hole in the closed field line region on the tailside (e.g. sphere with a handle attached). Topologically, the surface attains

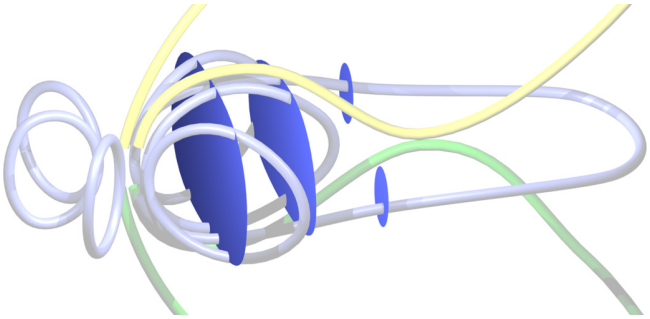

Fig. 6. The areas of closed magnetic field lines in three different x-planes added into Fig. 2.

genus one instead of the original genus zero. The space within that extra handle is called a plasmoid. It is possible for even multiple plasmoids to form (topological genus two or larger), but such structures did not form in the simulation of this event. Strictly speaking also a single plasmoid can have a topological genus larger than one: When a plasmoid has partially dissipated (Fig. 9f) the number of handles can be said to have increased although no additional plasmoids have formed.

The plasmoid search from the simulation results was conducted in the following way: Every cell in the simulation is classified by tracing the magnetic field line from the cell centroid. Four different topologies are possible: (1) closed, with both ends of the magnetic field line attached to the ionosphere, (2) open, with neither end attached to the ionosphere and (3)–(4) lobe field line, with one end attached to the ionosphere and the other not. After classification of the magnetic topology, the closed field line cells are identified at yz-planes starting from the plane $x = -10\ R_E$. A cluster of closed field line cells in a tail cross section is defined as a group of cells that share a vertex directly or through other cells in the same group (closed field lines cells). This is illustrated in Fig. 6 for three planes at different values of x. We define a plasmoid to begin at that value of x, where two or more such clusters also share a vertex through other closed field line cells downstream of that plane. In other words, a plasmoid is a singly connected 3-D set of closed field line cells, the 2-D intersection of which with some x-plane is multiply connected. Furthermore we define that a plasmoid constitutes the closed field line cells from the two or more clusters above and all other closed field line cells downstream that share a vertex directly with the clusters or through other closed field lines cells.

Although our definition of a plasmoid is different from Birn et al. (1989) and Farr et al. (2008), it does produce similar results. Our definition does not include free parameters and consequently the plasmoid search is straightforward to automate.

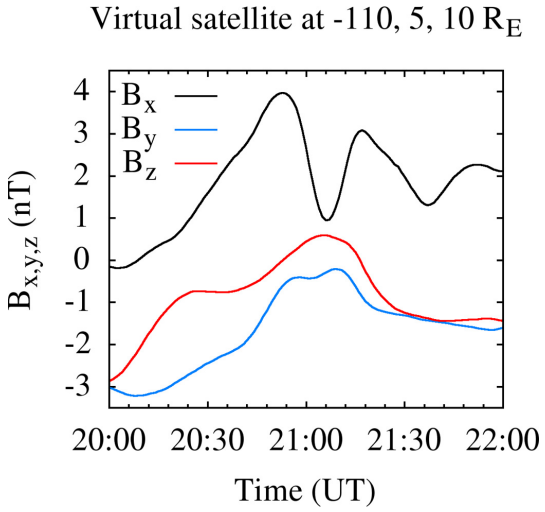

Fig. 7. The magnetic field observed in the simulation by a virtual satellite located at $(-110,\ 5,\ 10)\ R_E$ in GSE during and after the second AE intencification.

4.2 Plasmoid formation

The simulated event consisted of four intensifications of the AE index (starting times marked with vertical black lines in Fig. 4). During that period the IMF B_y and B_z underwent several rapid changes. The solar wind density increased by at least a factor of 2 during the first two AE intensifications. In the simulation a plasmoid forms after the 2nd and 4th AE intensifications. The vertical grey bars in Fig. 4 show the time period during which the closed magnetic field line region starts to extend further downstream and when the plasmoid finally dissipates. Figure 7 shows the bipolar B_z signature of the first plasmoid which formed around 21:00 UT as observed by a virtual satellite located at $(-110,\ 5,\ 10)\ R_E$ in GSE. The plasmoid signature between 20:30 and 21:30 UT in the simulation agrees with the one used by Moldwin and Hughes (1992): a bipolar signature in B_y and/or B_z not coincident with a neutral sheet crossing. Due to the small spatial resolution of the simulation in the tail the bipolar signature amplitude does not reach 3 nT.

Figures 8 to 10 show the magnetic topology classification in the simulation with the same color coding as in Fig. 2, with cells on a closed magnetic field line in blue and cells on a southern lobe field line in green. For clarity the northern lobe and IMF field line regions are not shown. Animation 1 shows the magnetic topology in the simulation with one minute intervals.

Figure 8 shows the magnetic field topology in the simulation during the first AE intensification at 16:31, 17:02, 17:21, 17:26, 18:02 and 18:25 UT. The IMF clock angle rotated from about $-130°$ at 16:40 UT to $10°$ at 17:00 UT and then returned slowly back to $-130°$ between 17:00 and 19:00 UT. Due to dayside reconnection, the most sunward lobe field line

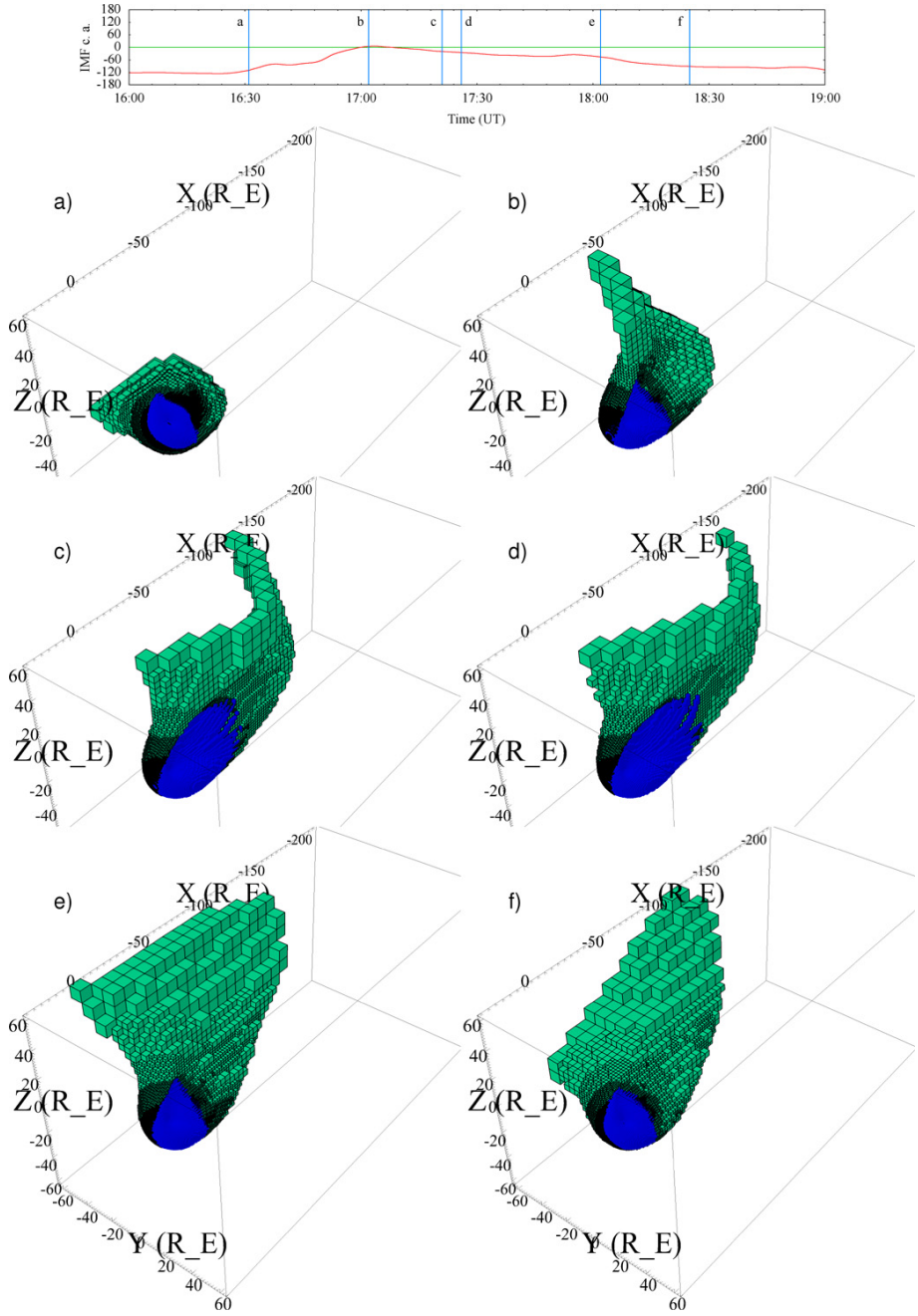

Fig. 8. Classification of cells in the simulation by their magnetic field topology is shown color coded. Blue cells show the region of closed magnetic field lines. Green cells show the region of southern lobe field lines. Northern lobe and IMF field line regions are not shown. The magnetic field topology in the simulation is shown during and after the first AE intencification at **(a)** 16:31, **(b)** 17:02, **(c)** 17:21, **(d)** 17:26, **(e)** 18:02 and **(f)** 18:25 UT. The IMF clock angle in each of the sub-panels is marked by blue bars at the top of the figure.

regions follow the direction of the IMF clock angle. This can be seen for southern lobe field lines in Figs. 8...10 as a green "wing" of lobe field lines on the dayside. In the case of positive IMF B_z, the southern lobe field lines drape over the closed field line region of the dayside magnetosphere towards the Northern Hemisphere, as seen at 17:02 and 17:21 UT in Fig. 8b–c. Figure 8 shows that changing the IMF clock angle increases the surface area of the lobe field line region perpendicular to the direction of solar wind bulk flow (yz-plane). The IMF rotation speed also affects how perpendicular the surface of the lobe field line region is to the solar wind bulk flow: faster rotation allows less time for the field lines to convect tailward thus producing a more perpendicular surface. At the same time the closed field

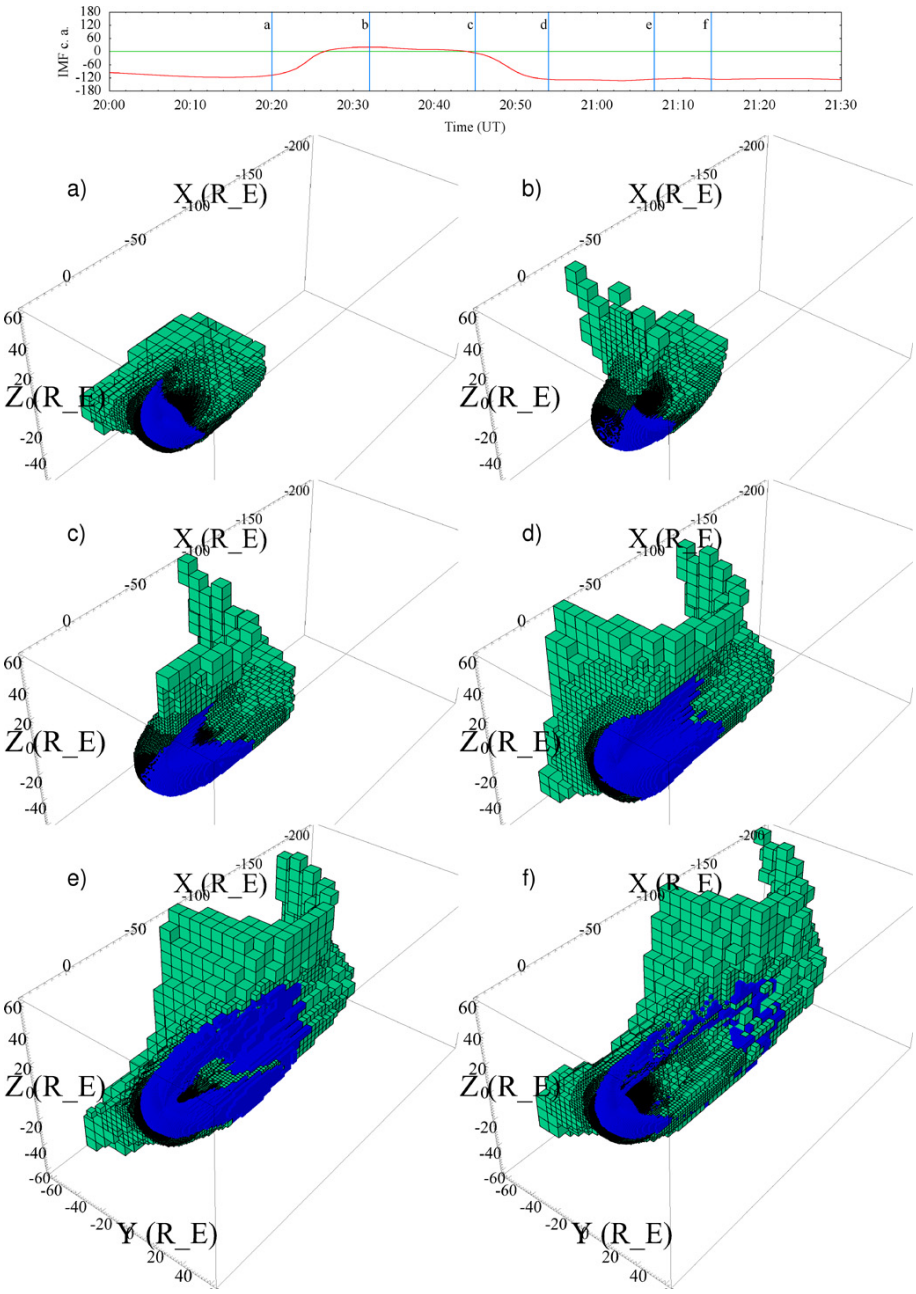

Fig. 9. Classification of cells in the simulation by their magnetic field topology is shown in the same format as in Fig. 8. The magnetic field topology and clock angle in the simulation are shown during and after the second AE intencification at **(a)** 20:20, **(b)** 20:32, **(c)** 20:45, **(d)** 20:54, **(e)** 21:07 and **(f)** 21:14 UT.

line region on the nightside of Earth extends downstream to about 50 R_E. Later (18:25 UT shown in Fig. 8f) the IMF clock angle slowly returns back to −130° between 17:00 and 19:00 UT and the closed field line region retreats back to about 20 R_E downstream. A common feature in global simulations using ideal MHD is that the reconnection line forms quite close to the Earth at about −20 R_E.

During the second AE intensification around 20:00 UT the IMF clock angle rotates rapidly from about −120° at 20:15 UT to 30° at 20:22 UT and 20 min later back to −130° between 20:38 and 20:45 UT. This IMF rotation can also be seen in Fig. 9, where the most sunward southern lobe field lines again follow the IMF clock angle. The format of Fig. 9 is the same as for Fig. 8 and shows six snapshots from the simulation at 20:20, 20:32, 20:45, 20:54, 21:07 and 21:14 UT. This AE intensification differs from the first one mainly in the IMF rotation speed, particularly because the IMF also returns to its original direction in less than 15 min. The average rotation speed during the second AE intensification is about 15° per minute. The fast and large rotations

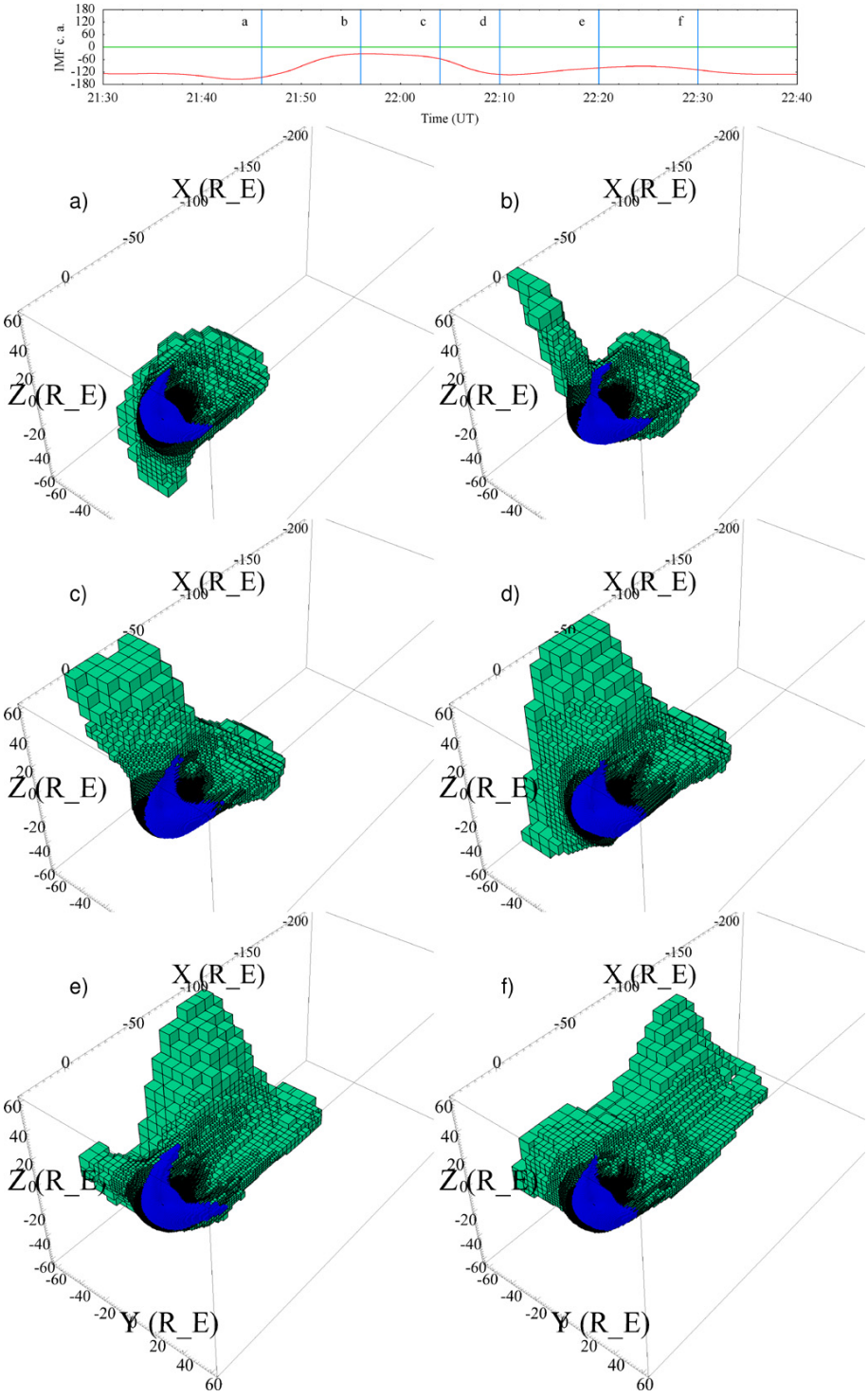

Fig. 10. Classification of cells in the simulation by their magnetic field topology is shown in the same format as in Fig. 8. The magnetic field topology and clock angle in the simulation are shown during and after the third AE intencification at **(a)** 21:46, **(b)** 21:56, **(c)** 22:04, **(d)** 22:10, **(e)** 22:20 and **(f)** 22:30 UT.

of the IMF clock angle during the second AE intensification create large surfaces of lobe field lines almost perpendicular to the direction of the solar wind bulk flow. These surfaces also move downstream with the solar wind similarly to the first AE intensification in Fig. 8. Figure 9c shows that during the initial clock-wise IMF rotation the closed field line

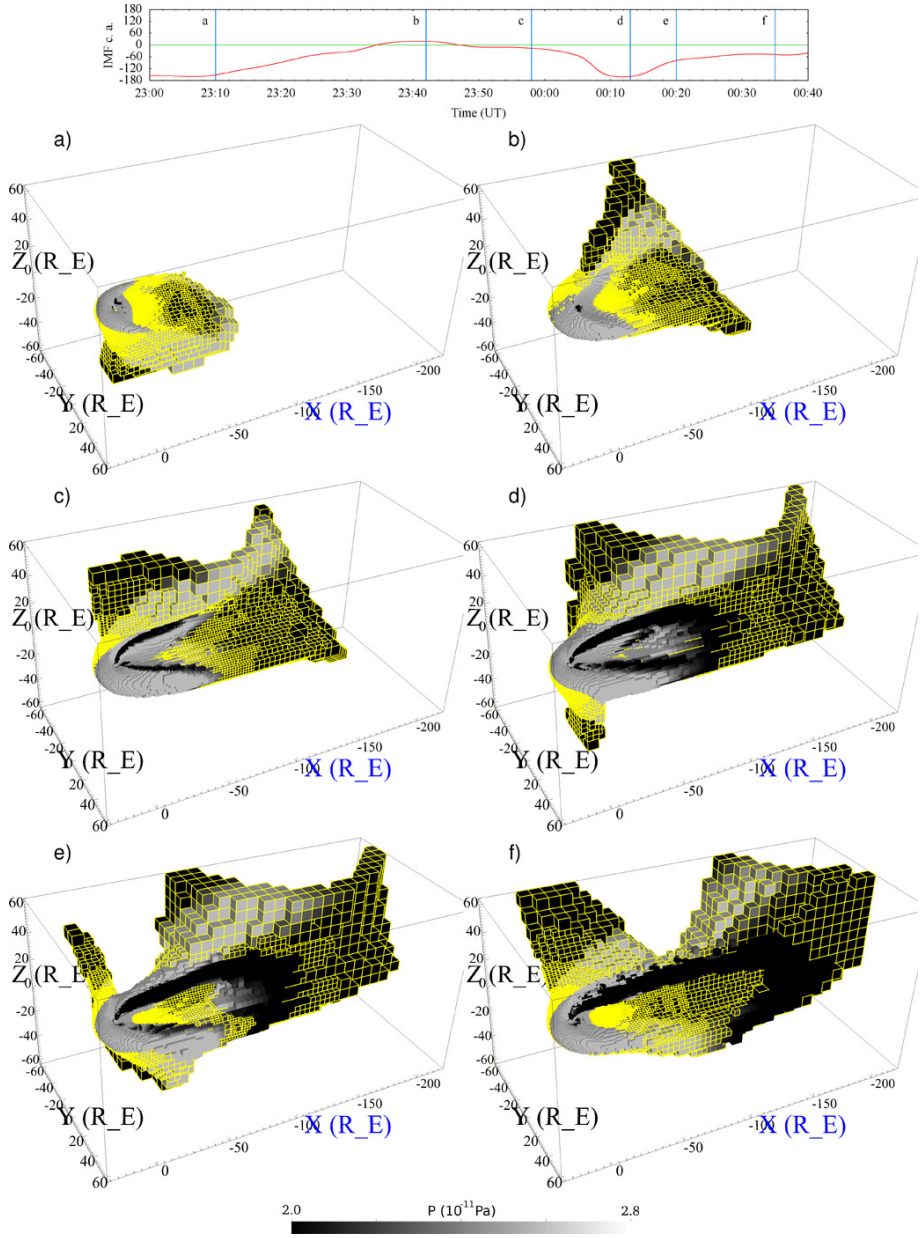

Fig. 11. Classification of cells in the simulation by their magnetic field topology is shown by the visible grid as in Fig. 8. Color coding shows the thermal pressure. The magnetic field topology, thermal pressure and clock angle in the simulation are shown during and after the fourth AE intencification at **(a)** 23:10, **(b)** 23:42, **(c)** 23:58, **(d)** 00:13, **(e)** 00:20 and **(f)** 00:35 UT.

region does not extend as far downstream as during the first AE intensification (Fig. 8c). In contrast, when the IMF returns quickly to its original direction during the second AE intensification, the closed field line region extends even further downstream (Fig. 9d). As the lobe field line region is transported downstream, the closed field line region finally detaches from the nightside of the Earth (Fig. 9e) forming a plasmoid. The plasmoid starts to form at 20:30 UT during the recovery phase of the substorm. The space between the Earth and the plasmoid, for example in Fig. 9e, is not topologically empty, but contains also northern lobe and IMF field lines which are not shown in the plot for clarity. In Fig. 9f the plasmoid has extended to about $-150\ R_E$ and has started to dissipate. Both Geotail and Wind are outside of the tail plasma sheet and do not show large plasmoid signatures during this event.

The third AE intensification around 22:00 UT differs from the previous two AE intensification in that the rotations of the IMF clock angle are smaller, only about 120°. The IMF rotation speed was of the same order as in the second

AE intensification, about 20° per minnute before and after 22:00 UT. Figure 10 shows the field line topologies during the third AE intensification at 21:46, 21:56, 22:04, 22:10, 22:20 and 22:30 UT in the same format as in Figs. 8 and 9. The dayside edge of the lobe field line region again follows the IMF clock angle. Figure 10 illustrates that a smaller rotation of the IMF creates a smaller surface of lobe field lines perpendicular to the solar wind bulk flow.

In other words, here the size of the surface perpendicular to solar wind bulk flow of lobe field lines depends only on the magnitude of the rotation of the IMF clock angle. This can be seen when comparing for example Figs. 9d and 10d. In Fig. 9d the perpendicular lobe field line surface spans nearly 180°, but in Fig. 10d it is noticeably smaller. Also the second jump in the IMF clock angle during the third AE intensification seen in Fig. 10e and f seems to smooth out the perpendicular surface even further. The closed field line region does not seem to expand downstream during the third AE intensification.

The last AE intensification of the period was at 23:00 UT, and its IMF characteristics mostly resembled those during the second AE intensification: Namely, the IMF clock angle rotated 190° between 23:07 and 23:30 UT with a speed of 8° per minute and returned only 15 min later to its original direction of −170° between 23:55 and 00:15 UT. Figure 11 shows the simulation at 23:10, 23:42, 23:58, 00:13, 00:20 and 00:35 UT. Color coding shows the thermal pressure in each cell. The topology of the magnetic field is classified as in previous figures. The region of southern lobe field lines is indicated by the grid colored yellow and the closed field line region is shown without the grid. A second plasmoid forms in the simulation during the last AE intensification. As previously, the most dayside lobe field lines follow the IMF clock angle. During the fourth AE intensification, the topology of the magnetic field evolves similarly to the second AE intensification. First the IMF rotates quickly from −170° at 23:07 UT to 20° at 20:30 UT and after about 15 min returns even faster to its original direction. Due to both rotations a large area of lobe magnetic field line is formed perpendicular to the solar wind bulk flow. Similarly to the second AE intensification, the lobe field line region moves downstream with the solar wind. The closed field line region on the night side also extends further downstream. As in Fig. 9 the region between the plasmoid and the Earth is not topologically empty, but contains also northern lobe and IMF field lines. Figure 11d shows the closed field line region just before it detaches from the night side of the Earth at 00:13 UT and Fig. 11e shows the formed plasmoid 7 min later. The color coding in Fig. 11f and e shows that the thermal pressure in the upstream part of the plasmoid is larger than in the downstream part, indicating that the plasmoid is pushed downstream by pressure gradients.

5 Plasmoid footpoints in the ionosphere

Next we investigate the ionospheric mapping of the plasmoid. First we find the plasmoid in the simulation as desribed in Sect. 4.1. Then the magnetic field is traced from every cell of the plasmoid into the inner boundary of the simulation. Figure 12 shows the plasmoid footpoints in the "ionosphere" at 4 R_E in red, other closed magnetic field lines in blue and northern lobe field lines in yellow. The same timesteps as in Fig. 9 viewed from about the same direction are shown in Fig. 12: 20:20, 20:32, 20:45, 20:54, 21:07 and 21:14 UT. Note how the rotating IMF distorts the polar cap boundary (Fig. 12c–d). Near the plasmoid footpoints the polar cap boundary shape is irregular (Fig. 12e). The ends of the plasmoid field lines map to the duskside of the Southern Hemisphere, where the shape of the polar cap boundary is also irregular (not shown). When the plasmoid has partially dissipated (Fig. 9f) the polar cap boundary in the simulation has returned to its original spherical shape as shown in Fig. 12f.

6 Discussion

In this paper we have presented GUMICS-4 simulation results of plasmoid formation during 18 February 2004. A plasmoid was defined as a major reconfiguration of the magnetotail where part of the tail plasma sheet is ejected downstream, in contrast to Earthward-propagating small plasmoids (see for example Zong et al., 2004). The event consists of four AE intensifications that occurred between 16:00 and 24:00 UT. During the event, GUMICS-4 reproduces in situ spacecraft observations quite well, indicating that the simulation results can be interpreted in light of physical processes in the near Earth space. We defined a plasmoid based only on the structure of the closed (connected to the Earth at both ends) magnetic field line region. We also presented an operational definition for these fully connected plasmoids that enable their automatic detection in simulations. The simulation results show the formation and detachment of a plasmoid after the second and fourth AE intensifications, while the other two AE intensifications were not associated with plasmoids. The plasmoid footpoints in the ionosphere form an irregularly shaped open-closed field line boundary.

Plasmoid formation is associated with sufficiently large and fast rotations of the IMF clock angle. This does not include such changes where one component of IMF is constant, e.g. northward turning of the IMF with constant B_y. The IMF rotation creates large surfaces of lobe field lines perpendicular to the solar wind flow, which also drape over the closed field line region of the dayside magnetosphere due to lobe reconnection (Figs. 9 and 11). As the lobe field line region outside of the magnetosphere moves downstream with the solar wind, the closed field line region inside of the magnetosphere extends along the flanks, perhaps due to a viscous interaction

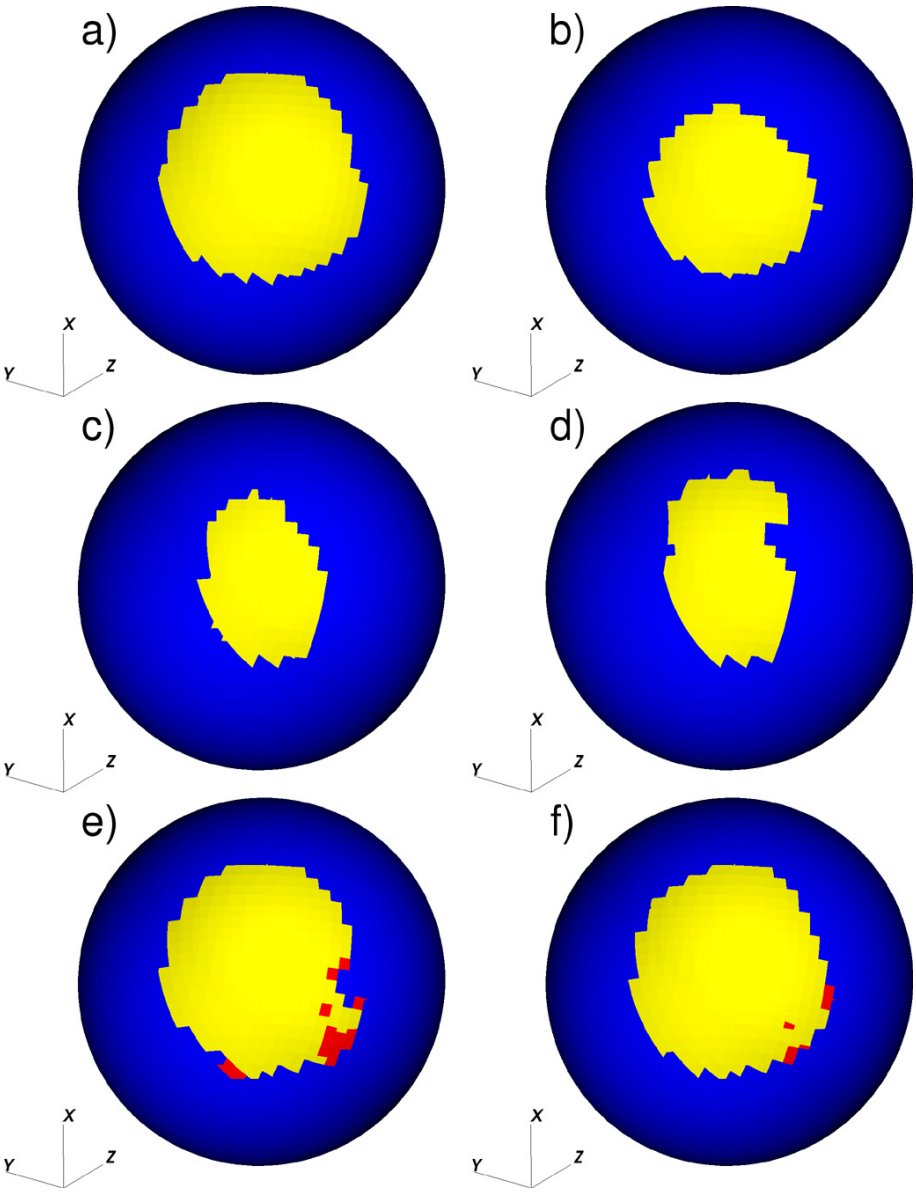

Fig. 12. The plasmoid footpoints at 4 R_E in the Northern Hemisphere in the simulation are shown in red, other closed magnetic field lines in blue and northern lobe field lines in yellow. The same timesteps are shown as in Fig. 9: **(a)** 20:20, **(b)** 20:32, **(c)** 20:45, **(d)** 20:54, **(e)** 21:07 and **(f)** 21:14 UT.

between the lobe and closed field lines. The lobe field lines also pass through the tail center directly downstream of the Earth (at $y = 0$ and $z = 0$) and, as a result of magnetic tension (due to the frozen-in condition between magnetic field and plasma), move downstream also in that region. Consequently the closed field line region also expands downstream from the Earth to fill the void left by the lobe field line region. By this mechanism the closed field line region eventually expands beyond $x = -100\ R_E$ downstream and reconnection around $x = -50\ R_E$ creates the plasmoid. Plasmoid formation seems to be initiated during lobe reconnection after a fast northward rotation of the IMF. The substorm onset for the second AE intensification coincides with an IMF B_z jump from -7 to -3 nT at 19:30 UT. In the simulation the large changes in IMF B_z between 19:30 and 20:00 UT are smeared out and lobe reconnection starts only when during the large rotation of the IMF around 20:20 UT.

The literature lists several plasmoid features that are consistent with the results reported here: Hones et al. (1984b) reported on plasmoids of sizes from 75 to 150 R_E. Moldwin and Hughes (1992) surveyed 366 plasmoidlike structures and the events most comparable to our results are the isolated plasmoids with sizes up to 80 R_E. The larger of the two plasmoids in the GUMICS-4 simulation of this event is

about 70 R_E in the x-direction. Furthermore, Moldwin and Hughes (1992) observed that the size and speed of plasmoids as a function of distance downtail does not change beyond 100 R_E. This is in agreement with the simulation results, as the size and speed of the plasmoids were roughly constant after formation near $x_{GSE} = -50\ R_E$ until their dissipation. The speed of the plasmoids reported here are close to the solar wind velocity. The observations of Moldwin and Hughes (1992) also suggest that plasmoids are open magnetic structures, possibly flux ropes, and not closed loops of magnetic field lines. In our simulation, the plasmoids form tailward of the reconnection line, while keeping some field lines attached to the Earth until dissipation. Nagai et al. (1997) concluded that there are two sites preferable for magnetic reconnection: inside $x_{GSM} = -30\ R_E$ and near $x_{GSM} = -100\ R_E$. In our simulation plasmoids are formed by reconnection under suitable IMF conditions around $x_{GSE} = -50\ R_E$ (Figs. 9c and d, 11c and d). After formation the plasmoids propagate downstream and are dissipated by reconnection near $x_{GSE} = -100\ R_E$ (Fig. 9f) or beyond $x_{GSE} = -150\ R_E$ (Fig. 11f).

Pulkkinen et al. (1998) examined two sequential substorm onsets, of which the first one occurred during persistently negative IMF B_z while the second one was associated with a northward turning of B_z. They observed that while the first onset remained localized, the second onset led to a major reconfiguration of the magnetotail. These observations are qualitatively in agreement with our simulation, which shows that a fast and large rotation of the IMF is required for plasmoid formation and launch, thus leading to a reconfiguration of the tail. Furthermore, according to Ieda at al. (2001), plasmoid formation is associated with the IMF first turning southward and then about 20 min later returning to northward. Here we report that the simulation plasmoids are formed when the delay between IMF rotations is from about 20 to 40 min.

Kivelson et al. (1996) performed global MHD simulations using artificial solar wind as the simulation input, and showed that for a negative IMF B_y the flux rope closed field lines link the northern morning ($y < 0$) ionosphere to the southern evening ($y > 0$) ionosphere. This is also the case in our simulation, where IMF B_y is negative for at least an hour before the plasmoids form. In another global MHD simulation, Slinker et al. (1995) observed the formation of a plasmoid by southward turning of the IMF after a long period of steady northward IMF. Similarly to the results presented here, the formation region of plasmoids was at about $x = -45\ R_E$ and the plasmoid velocity approached the solar wind velocity.

Farr et al. (2008) presented LFM global MHD simulation (Lyon et al., 2004) results for a pair of substorms on 11 August 2002. A flux tube formed in the simulation during southward B_z, following a north-south rotation about half an hour earlier. They mapped the flux tube into the ionosphere, and similarly to Kivelson et al. (1996), found that for positive IMF B_y the flux rope linked the northern dusk ionosphere to the southern dawn ionosphere. Furthermore, the ionospheric mapping showed a non-trivial structure of the flux rope inside of the closed field line region, in agreement with the results presented here.

The results presented here can also explain in part the close association observed between substorms and plasmoids (Moldwin and Hughes, 1993). Hsu and McPherron (2002) showed that half of all substorms are triggered by northward turning of the IMF. They did not, however, investigate simultaneous change in B_y during northward turning, which could have resulted in large rotation of the IMF instead. Based on our results this rotation could also create a plasmoid, especially if the IMF subsequently rotates again. Thus in some cases neither a substorm nor a plasmoid would result from the other, but instead they would be both the result of a fast rotation or rotations of the IMF clock angle.

Our main result in this paper is that the formation of plasmoids in the simulation is the result of two consecutive, sufficiently large and fast rotations of the IMF clock angle. As a result of these rotations the lobe field line region moving with the solar wind reduces the thermal pressure at the downstream edge of the closed field line region directly behind the Earth. Consequently the closed field line region extends downstream due to this pressure gradient, initiating the formation of a plasmoid. On the other hand when either the rotation speed of the IMF clock angle or the angle of rotation are small, the resulting pressure gradient is much smaller or nonexistent and a plasmoid does not form.

Supplementary material related to this article is available online at: http://www.ann-geophys.net/29/167/2011/angeo-29-167-2011-supplement.zip.

Acknowledgements. This work is a part of the project 200141-QuESpace, funded by the European Research Council under the European Community's seventh framework programme. The work of IH and MP is supported by project 218165 of the Academy of Finland. We thank the Cluster Active Archive, the FGM team and the instrument PI E. A. Lucek for providing freely the Cluster data. We also thank the Coordinated Data Analysis Web, the instrument PI N. Ness for providing the ACE data and the instrument PI H. Singer for providing the GOES data used in this study. Finally we thank the World Data Center for Geomagnetism for the Kyoto AE index service.

Topical Editor R. Nakamura thanks J. Birn and another anonymous referee for their help in evaluating this paper.

References

Balogh, A., Carr, C. M., Acuña, M. H., Dunlop, M. W., Beek, T. J., Brown, P., Fornacon, K.-H., Georgescu, E., Glassmeier, K.-H., Harris, J., Musmann, G., Oddy, T., and Schwingenschuh, K.: The Cluster Magnetic Field Investigation: overview of in-flight

performance and initial results, Ann. Geophys., 19, 1207–1217, doi:10.5194/angeo-19-1207-2001, 2001.

Birn, J., Hesse, M., and Schindler, K.: Filamentary Structure of a Three-Dimensional Plasmoid, J. Geophys. Res., 94, 241–251, 1989.

Brackbill, J. U. and Barnes, D. C.: The Effect of Nonzero $\nabla\cdot\mathbf{B}$ on the Numerical Solution of the Magnetohydrodynamic Equations, J. Comput. Phys., 35, 426–430, 1980.

Coppi, B., Laval, G., and Pellat, R.: Dynamics of the Geomagnetic Tail, Phys. Rev. Lett., 16, 1207–1210, 1966.

Farr, N. L., Baker, D. N., and Wiltberger, M.: Complexities of a 3-D plasmoid flux rope as shown by an MHD simulation, J. Geophys. Res., 113, A12202, doi:10.1029/2008JA013328, 2008.

Hones Jr., E. W.: Substorm Processes in the Magnetotail: Comments on "On Hot Tenuous Plasmas, Fireballs, and Boundary Layers in the Earth's Magnetotail" by L. A. Frank, K. L. Ackerson, and R. P. Lepping, J. Geophys. Res., 82, 5633–5640, 1977.

Hones Jr., E. W.: Transient phenomena in the magnetotail and their relation to substorms, Space Sci. Rev., 23, 393–410, 1979.

Hones Jr., E. W., Baker, D. N., Bame, S. J., Feldman, W. C., Gosling, J. T., McComas, D. J., Zwickl, R. D., Slavin, J. A., Smith, E. J., and Tsurutani, B. T.: Structure of the magnetotail at 220 R_{E} and its response to geomagnetic activity, Geophys. Res. Lett., 11, 5–7, 1984.

Hones Jr, E. W., Birn, J., Baker, D. N., Bame, S. J., Feldman, W. C., McComas, D. J., Zwickl, R. D., Slavin, J. A., Smith, E. J., and Tsurutani, B. T.: Detailed examination of a plasmoid in the distant magnetotail with ISEE 3, Geophys. Res. Lett., 11, 1046–1049, 1984.

Hughes, W. J. and Sibeck, D. G.: On the 3-dimensional structure of plasmoids, Geophys. Res. Lett., 14, 636–639, 1987.

Hsu, T.-S. and McPherron, R. L.: An evaluation of the statistical significance of the association between northward turnings of the interplanetary magnetic field and substorm expansion onsets, J. Geophys. Res., 107, A11, doi:10.1029/2000JA000125, 2002.

Ieda, A., Machida, S., Mukai, T., Saito, Y., Yamamoto, T., Nishida, A., Terasawa, T., and Kokubun. S.: Statistical analysis of the plasmoid evolution with Geotail observations, J. Geophys. Res., 103, 4453–4465, 1998.

Ieda, A., Fairfield, D. H., Mukai, T., Saito, Y., Kokubun, S., Liou, K., Meng, C.-I., Parks, G. K., and Brittnacher, M. J.: Plasmoid ejection and auroral brightenings, J. Geophys. Res., 106, 3845–3857, 2001.

Janhunen, P.: GUMICS-3 – a global ionosphere-magnetosphere coupling simulation with high ionospheric resolution, Proceedings of Environmental Modelling for Space-Based Applications, 18–20 September 1996, Eur. Space Agency Spec. Publ., ESA SP-392, available at: http://www.space.fmi.fi/~pjanhune/papers/ESTEC96/ESTEC96.ps.gz, 1996.

Janhunen, P., Koskinen, H. E. J., and Pulkkinen, T. I.: A new global ionosphere-magnetosphere coupling simulation utilizing locally varying time step, Proceeding of Third International Conference on Substorms, Versailles, France, 12–17 May 1996, ESA SP-389, 1996.

Kivelson, M. G., Khurana, K. K., Walker, R. J., Kepko, L., and Xu, D.: Flux ropes, interhemispheric conjugacy, and magnetospheric current closure, J. Geophys. Res., 101, 27341–27350, 1996.

Lyon, J. G., Fedder, J. A., and Mobarry, C. M.: The Lyon-Fedder-Mobarry (LFM) global MHD magnetospheric simulation code, J. Atmos. Solar-Terr. Phy., 66, 1333–1350, 2004.

McPherron, R. L.: Physical Processes Producing Magnetospheric Substorms and Magnetic Storms, Geomagnetism vol. 4, J. Jacobs, Academic Press Ltd., London, England, 593-739, available at: http://geomag.usgs.gov/iagaxiii/papers/McPherron_1991_JacobsBook.pdf, 1991.

Moldwin, M. B. and Hughes, W. J.: On the Formation and Evolution of Plasmoids: A Survey of ISEE 3 Geotail Data, J. Geophys. Res., 97, 19259–19282, 1992.

Moldwin, M. B. and Hughes, W. J.: Geomagnetic Substorm Association of Plasmoids, J. Geophys. Res., 98, 81–88, 1993.

Nagai, T., Nakamura, R., Mukai, T., Yamamoto, T., Nishida, A., and Kokubun, S.: Substorms, tail flows and plasmoids, Adv. Space Res., 20, 961–971, 1997.

Palmroth, M., Pulkkinen, T. I., Janhunen, P., and Wu, C.-C.: Stormtime energy transfer in global MHD simulation, J. Geophys. Res., 108, 1048, doi:10.1029/2002JA009446, 2003.

Palmroth, M., Koskinen, H. E. J., Pulkkinen, T. I., Toivanen, P. K., Janhunen, P., Milan, S. E., and Lester, M.: Magnetospheric feedback in solar wind energy transfer, J. Geophys. Res., 115, A00I10, doi:10.1029/2010JA015746, available at: http://www.agu.org/journals/ja/ja1012/2010JA015746/, 2010.

Pulkkinen, T. I., Baker, D. N., Frank, L. A., Sigwarth, J. B., Opgenoorth, H. J., Greenwald, R., Friis-Christensen, E., Mukai, T., Nakamura, R., Singer, H., Reeves, G. D., and Lester, M.: Two substorm intensifications compared: Onset, expansion, and global consequences, J. Geophys. Res., 103, 15–27, 1998.

Silbergleit, V. M., Zossi de Artigas, M. M., and Manzano J. R.: Energy dissipation in substorms: plasmoids ejection, J. Atmos. Solar-Terr. Phy., 59, 1355–1358, 1997.

Slavin, J. A., Daly, P. W., Smith, E. J., Sanderson, T. R., Wenzel, K.-P., Lepping, R. P., and Kroehl, H. W.: Magnetic configuration of the distant plasma sheet: ISEE 3 observations, Magnetotail Physics, Lui, A. T. Y., The Johns Hopkins University Press, Baltimore, 59–63, 1987.

Slinker, S. P., Fedder, J. A., and Lyon, J. G.: Plasmoid formation and evolution in a numerical simulation of a substorm, Geophys. Res. Lett., 22, 859–862, 1995.

Zong, Q.-G., Fritz, T. A., Pu, Z. Y., Fu, S. Y., Baker, D. N., Zhang, H., Lui, A. T., Vogiatzis, I., Glassmeier, K.-H., Korth, A., Daly, P. W., Balogh, A., and Reme, H.: Cluster observations of earthward flowing plasmoid in the tail, Geophys. Res. Lett., 31, L18803, doi:10.1029/2004GL020692, 2004.

Finnish Meteorological Institute Contributions

1. Joffre, Sylvain M., 1988. Parameterization and assessment of processes affecting the long-range transport of airborne pollutants over the sea. 49 p.

2. Solantie, Reijo, 1990. The climate of Finland in relation to its hydrology, ecology and culture. 130 p.

3. Joffre, Sylvain M. and Lindfors, Virpi, 1990. Observations of airborne pollutants over the Baltic Sea and assessment of their transport, chemistry and deposition. 41 p.

4. Lindfors, Virpi, Joffre, Sylvain M. and Damski, Juhani, 1991. Determination of the wet and dry deposition of sulphur and nitrogen compounds over the Baltic Sea using actual meteorological data. 111 p.

5. Pulkkinen, Tuija, 1992. Magnetic field modelling during dynamic magnetospheric processes. 150 p.

6. Lönnberg, Peter, 1992. Optimization of statistical interpolation. 157 p.

7. Viljanen, Ari, 1992. Geomagnetic induction in a one- or two-dimensional earth due to horizontal ionospheric currents. 136 p.

8. Taalas, Petteri, 1992. On the behaviour of tropospheric and stratospheric ozone in Northern Europe and in Antarctica 1987-90. 88 p.

9. Hongisto, Marke, 1992. A simulation model for the transport, transformation and deposition of oxidized nitrogen compounds in Finland — 1985 and 1988 simulation results. 114 p.

10. Taalas, Petteri, 1993. Factors affecting the behaviour of tropospheric and stratospheric ozone in the European Arctic and Antarctica. 138 s.

11. Mälkki, Anssi, 1993. Studies on linear and non-linear ion waves in the auroral acceleration region. 109 p.

12. Heino, Raino, 1994. Climate in Finland during the period of meteorological observations. 209 p.

13. Janhunen, Pekka, 1994. Numerical simulations of E-region irregularities and ionosphere-magnetosphere coupling. 122 p.

14. Hillamo, Risto E., 1994. Development of inertial impactor size spectroscopy for atmospheric aerosols. 148 p.

15. Pakkanen, Tuomo A., 1995. Size distribution measurements and chemical analysis of aerosol components. 157 p.

16. Kerminen, Veli-Matti, 1995. On the sulfuric acid-water particles via homogeneous nucleation in the lower troposphere. 101 p.

17. Kallio, Esa, 1996. Mars-solar wind interaction: Ion observations and their interpretation. 111 p.

18. Summanen, Tuula, 1996. Interplanetary Lyman alpha measurements as a tool to study solar wind properties. 114 p.

19. Rummukainen, Markku, 1996. Modeling stratospheric chemistry in a global three-dimensional chemical transport model, SCTM-1. Model development. 206 p.

20. Kauristie, Kirsti, 1997. Arc and oval scale studies of auroral precipitation and electrojets during magnetospheric substorms. 134 p.

21. Hongisto, Marke, 1998. Hilatar, A regional scale grid model for the transport of sulphur and nitrogen compounds. 152 p.

22. Lange, Antti A.I., 1999. Statistical calibration of observing systems. 134 p.

23. Pulkkinen, Pentti, 1998. Solar differential rotation and its generators: computational and statistical studies. 108 p.

24. Toivanen, Petri, 1998. Large-scale electromagnetic fields and particle drifts in time-dependent Earth's magnetosphere. 145 p.

25. Venäläinen, Ari, 1998. Aspects of the surface energy balance in the boreal zone. 111 p.

26. Virkkula, Aki, 1999. Field and laboratory studies on the physical and chemical properties of natural and anthropogenic tropospheric aerosol. 178 p.

27. Siili, Tero, 1999. Two-dimensional modelling of thermal terrain-induced mesoscale circulations in Mars' atmosphere. 160 p.

28. Paatero, Jussi, 2000. Deposition of Chernobyl-derived transuranium nuclides and short-lived radon-222 progeny in Finland. 128 p.

29. Jalkanen, Liisa, 2000. Atmospheric inorganic trace contaminants in Finland, especially in the Gulf of Finland area. 106 p.

30. Mäkinen, J. Teemu, T. 2001. SWAN Lyman alpha imager cometary hydrogen coma observations. 134 p.

31. Rinne, Janne, 2001. Application and development of surface layer flux techniques for measurements of volatile organic compound emissions from vegetation. 136 p.

32. Syrjäsuo, Mikko T., 2001. Auroral monitoring system: from all-sky camera system to automated image analysis. 155 p.

33. Karppinen, Ari, 2001. Meteorological pre-processing and atmospheric dispersion modelling of urban air quality and applications in the Helsinki metropolitan area. 94 p.

34. Hakola, Hannele, 2001. Biogenic volatile organic compound (VOC) emissions from boreal deciduous trees and their atmospheric chemistry. 125 p.

35. Merenti-Välimäki, Hanna-Leena, 2002. Study of automated present weather codes.153 p.

36. Tanskanen, Eija I., 2002. Terrestrial substorms as a part of global energy flow. 138 p.

37. Nousiainen, Timo, 2002. Light scattering by nonspherical atmospheric particles. 180 p.

38. Härkönen, Jari, 2002. Regulatory dispersion modelling of traffic-originated pollution. 103 p.

39. Oikarinen, Liisa, 2002. Modeling and data inversion of atmospheric limb scattering measurements. 111 p.

40. Hongisto, Marke, 2003. Modelling of the transport of nitrogen and sulphur contaminants to the Baltic Sea Region. 188 p.

41. Palmroth, Minna, 2003. Solar wind – magnetosphere interaction as determined by observations and a global MHD simulation. 147 p.

42. Pulkkinen, Antti, 2003. Geomagnetic induction during highly disturbed space weather conditions: Studies of ground effects 164 p.

43. Tuomenvirta, Heikki, 2004. Reliable estimation of climatic variations in Finland. 158 p.

44. Ruoho-Airola, Tuija, 2004. Temporal and regional patterns of atmospheric components affecting acidification in Finland. 115 p.

45. Partamies, Noora, 2004. Meso-scale auroral physics from groundbased observations. 122 p.

46. Teinilä, Kimmo, 2004. Size resolved chemistry of particulate ionic compounds at high latitudes. 138 p.

47. Tamminen, Johanna, 2004. Adaptive Markov chain Monte Carlo algorithms with geophysical applications. 156 p.

48. Huttunen, Emilia, 2005. Interplanetary shocks, magnetic clouds, and magnetospheric storms. 142 p.

49. Sofieva, Viktoria, 2005. Inverse problems in stellar occultation. 110 p.

50. Harri, Ari-Matti, 2005. In situ observations of the atmospheres of terrestrial planetary bodies. 246 p.

51. Aurela, Mika, 2005. Carbon dioxide exchange in subarctic ecosystems measured by a micrometeorological technique. 132 p.

52. Damski, Juhani, 2005. A Chemistry-transport model simulation of the stratospheric ozone for 1980 to 2019. 147 p.

53. Tisler, Priit, 2006. Aspects of weather simulation by numerical process. 110 p.

54. Arola, Antti, 2006. On the factors affecting short- and long-term UV variability. 82 p.

55. Verronen, Pekka T., 2006. Ionosphere-atmosphere interaction during solar proton events. 146 p.

56. Hellén, Heidi, 2006. Sources and concentrations of volatile organic compounds in urban air. 134 p.

57. Pohjola, Mia, 2006. Evaluation and modelling of the spatial and temporal variability of particulate matter in urban areas. 143 p.

58. Sillanpää, Markus, 2006. Chemical and source characterisation of size-segregated urban air particulate matter. 184 p.

59. Niemelä, Sami, 2006. On the behaviour of some physical parameterization methods in high resolution numerical weather prediction models. 136 p.

60. Karpechko, Alexey, 2007. Dynamical processes in the stratosphere and upper troposphere and their influence on the distribution of trace gases in the polar atmosphere. 116 p.

61. Eresmaa, Reima, 2007. Exploiting ground-based measurements of Global Positioning System for numerical weather prediction. 95 p.

62. Seppälä, Annika, 2007. Observations of production and transport of NOx formed by energetic particle precipitation in the polar night atmosphere. 100 p.

63. Rontu, Laura, 2007. Studies on orographic effects in a numerical weather prediction model. 151 p.

64. Vajda, Andrea, 2007. Spatial variations of climate and the impact of disturbances on local climate and forest recovery in northern Finland. 139 p.

65. Laitinen, Tiera, 2007. Rekonnektio Maan magnetosfäärissä – Reconnection in Earth's magnetosphere. 226 s.

66. Vanhamäki, Heikki, 2007. Theoretical modeling of ionospheric electrodynamics including induction effects. 170 p.

67. Lindfors. Anders, 2007. Reconstruction of past UV radiation. 123 p.

68. Sillanpää, Ilkka, 2008. Hybrid modelling of Titan's interaction with the magnetosphere of Saturn. 200 p.

69. Laine, Marko, 2008. Adaptive MCMC methods with applications in environmental and geophysical models. 146 p.

70. Tanskanen, Aapo, 2008. Modeling of surface UV radiation using satellite data. 109 p.

71. Leskinen, Ari, 2008. Experimental studies on aerosol physical properties and transformation in environmental chambers. 116 p.

72. Tarvainen, Virpi, 2008. Development of biogenetic VOC emission inventories for the boreal forest. 137 p.

73. Lohila, Annalea, 2008. Carbon dioxide exchange on cultivated and afforested boreal peatlands. 110 p.

74. Saarikoski, Sanna, 2008. Chemical mass closure and source-specific composition of atmospheric particles. 182 p.

75. Pirazzini, Roberta, 2008. Factors controlling the surface energy budget over snow and ice. 141 p.

76. Salonen, Kirsti, 2008. Towards the use of radar winds in numerical weather prediction. 87 p.

77. Luojus, Kari, 2009. Remote sensing of snow-cover for the boreal forest zone using microwave radar. 178 p.

78. Juusola, Liisa, 2009. Observations of the solar wind-magnetosphere-ionosphere coupling. 167 p.

79. Waldén, Jari, 2009. Meteorology of gaseous air pollutants. 177 p.

80. Mäkelä, Jakke, 2009. Electromagnetic signatures of lightning near the HF frequency band. 152 p.

81. Thum, Tea, 2009. Modelling boreal forest CO_2 exchange and seasonality. 140 p.

82. Lallo, Marko, 2010. Hydrogen soil deposition and atmospheric variations in the boreal zone. 91 p.

83. Sandroos, Arto, 2010. Shock acceleration in the solar corona. 116 p.

84. Lappalainen, Hanna, 2010. Role of temperature in the biological activity of a boreal forest. 107 p.

85. Mielonen, Tero, 2010. Evaluation and application of passive and active optical remote sensing methods for the measurement of atmospheric aerosol properties. 125 p.

86. Lakkala, Kaisa, 2010. High quality polar UV measurements : scientific analyses and transfer of the irradiance scale. 156 p.

87. Järvinen, Riku, 2011. On ion escape from Venus. 150 p.

88. Saltikoff, Elena, 2011. On the use of weather radar for mesoscale applications in northern conditions. 120 p.

89. Timonen, Hilkka, 2011. Chemical characterization of urban background aerosol using online and filter methods. 176 p.

90. Hyvärinen, Otto, 2011. Categorical meteorological products : evaluation and analysis.138 p.

91. Mäkelä, Antti, 2011. Thunderstorm climatology and lightning location applications in northern Europe. 158 p.

92. Hietala, Heli, 2012. Multi-spacecraft studies on space plasma shocks. 132 p.

93. Leinonen, Jussi, 2013. Thunderstorm climatology and lightning location applications in northern Europe. 140 p.

94. Gregow, Hilppa, 2013. Impact of strong winds, heavy snow loads and soil frost conditions on the risks to forests in Northern Europe. 178 p.

95. Henriksson, Svante, 2013. Modeling key processes causing climate change and variability. 98 p.

96. Valkonen, Teresa, 2013. Surface influence on the marine and coastal Antarctic atmosphere. 114 p

97. Saarnio, Karri, 2013. Chemical characterisation of fine particles from biomass burning. 180 p.

98. Honkonen, Ilja, 2013. Development, validation and application of numerical space environment models. 148 p.