Analysis of the solver performance for stokes flow problems in glass forming process simulation models

Groot, J.A.W.M.

Award date:
2007

TECHNISCHE UNIVERSITEIT EINDHOVEN
Department of Mathematics and Computer Science

Analysis of the Solver Performance for Stokes Flow Problems in Glass Forming Process Simulation Models

by

J. A. W. M. Groot

Supervisors:

Prof. dr. R. M. M. Mattheij (TU/e)
Dr. C. G. Giannopapa (TU/e)
Dr. S. W. Rienstra (TU/e)
Dr. D. Hegen (TNO Science and Technology)

Eindhoven, Jan 2007

Abstract

During a glass forming process in industrial glass manufacturing, a glass gob that comes directly from the furnace is forced into a desired shape. In general, practical experiments with glass forming are considerably expensive, whereas the majority has to be performed under complicated circumstances, e.g. high temperatures. In addition, the glass industry is fairly secretive about experimental data. Therefore, computer simulation models are required to gain a better understanding and improvement of glass forming processes.

The glass forming process simulation models considered revolve around three coupled, principal physical problems. These are successively a flow problem for the motion of glass and air, an energy problem for the energy exchange in glass, air and equipment, and an interface problem for the location of the interfaces between glass and air. The boundary value problems are discretised by means of finite element methods and a suitable time discretisation scheme. Subsequently, an iterative solver with preconditioning is applied to solve the resulting systems of equations for each successive time step.

Unfortunately, problems regarding the solver performance occur for flow problems in TNO Glass Group's glass forming process simulation models, as application of mesh refinement produces an excessive increase in the number of iterations required by the iterative solvers for convergence, which finally results in termination of the solvers. This solver problem does not occur, or at least to a lesser degree, for the energy problem and the level set problem.

The solver performance is examined for TNO Glass Group's glass pressing process simulation model. Since the solver is relatively slow and instable for the three dimensional simulation model, only the axi-symmetric model is used to test the solver performance. The discretisation methods used for flow problems in the simulation model are validated.

In TNO Glass Group's pressing process simulation model, ILU preconditioning with Sloan reordering of the unknowns is used to improve convergence of iterative solvers. ILU preconditioning can be improved by allowing additional fill-in and using Cuthill Mc Kee ordering instead of Sloan ordering. The resulting improvement of the solver performance in TNO Glass Group's axi-symmetric pressing process simulation model is formidable. In addition it is shown that ILU preconditioning is superior to several other preconditioners, such as Gauss-Seidel and Eisenstat. Other preconditioners such as multigrid methods are suggested, but not tested due to implementation issues.

Acknowledgements

In the first place, I would like to express my sincere gratitude to my supervisors dr. D. Hegen (TNO Science and Technology), dr. C. G. Giannopapa, prof. dr. R. M. M. Mattheij and dr. S. W. Rienstra for their continuous guidance, considerable interest and invaluable motivation. I am as well sincerely grateful to dr. J. M. L. Maubach for his aid in the finite element analysis of Stokes flow problems. Furthermore, I am grateful to my colleagues during my work placement at TNO Science and Technology for their involvement with my final project and all valuable experiences I have had at TNO. Finally, I would like to thank Dr. Guus Segal for his responsive assistance with Sepran.

About TNO

The final project that is subject of this Master's thesis has been carried out at TNO Science and Technology in Eindhoven. TNO is the Dutch organisation for applied technological and scientific research and exists by Dutch law. TNO is a country-wide organisation with approximately 4500 employees and many offices, such as in Eindhoven, Helmond, Delft, Apeldoorn, Enschede and Den Helder. TNO is organised in five main areas, as can be seen in Figure 1.

One of these areas is TNO Science and Technology, which focusses on materials technology, process innovation, product development and process modelling, in view of the six main markets represented in Figure 1. Currently, about 1100 experts are employed at TNO Science and Technology. The main locations of TNO Science and Technology are Delft and Eindhoven.

This final project about free surface flow calculations in glass forming process simulation models is part of a larger project of TNO Glass Group. TNO Glass Group is a department of TNO Science and Technology that forms a centre of expertise towards the glass manufacturing industry. The glass forming process simulation models that this thesis is concerned with are designed by TNO Glass Group.

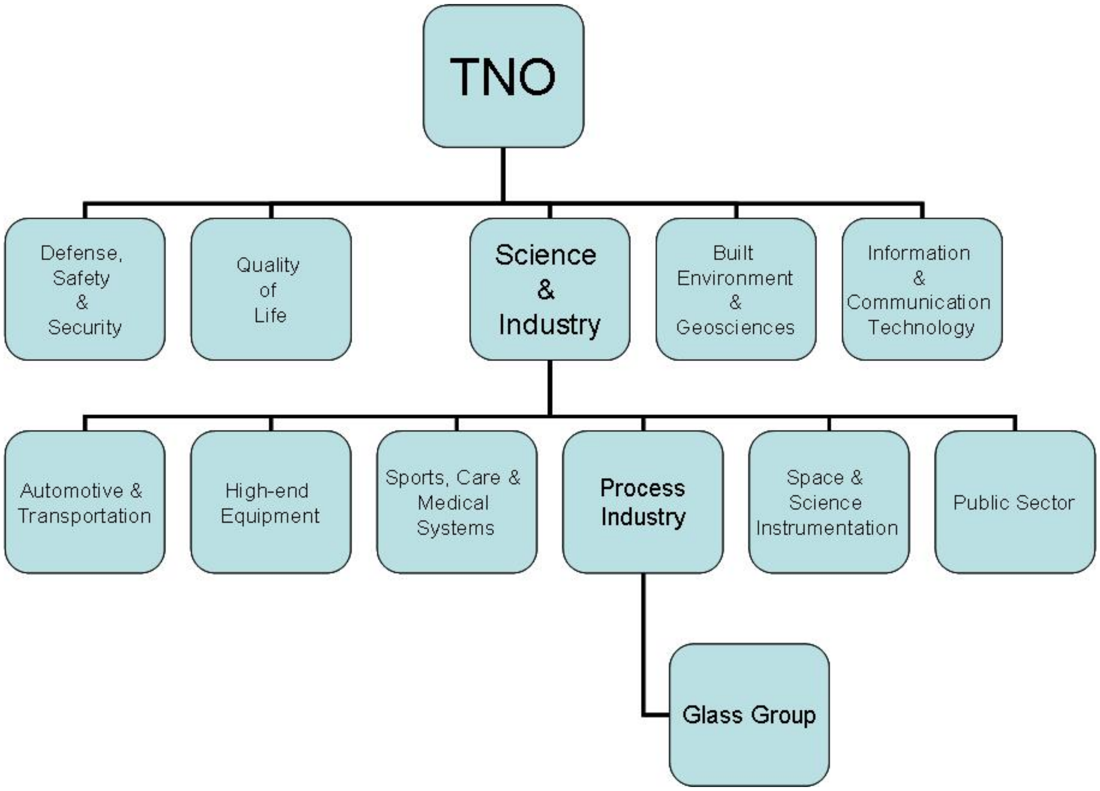

TNO
Defense, Safety & Security
Quality of Life
Science & Industry
Built Environment & Geosciences
Information & Communication Technology
Automotive & Transportation
High-end Equipment
Sports, Care & Medical Systems
Process Industry
Space & Science Instrumentation
Public Sector
Glass Group

Figure 1: The organisation of TNO and the potential markets of TNO Science and Technology

Nomenclature

Abbreviations

Symbol		Meaning
BC	:	Boundary Condition
BVP	:	Boundary Value Problem
FEM	:	Finite Element Method
ILU	:	Incomplete LU factorisation
PDE	:	Partial Differential Equation

Arithmetic

Symbol		Description	Mathematical Definition
$\cdot$	:	column inner product	$\forall_{\boldsymbol{u}\in\mathbb{R}^n,\ \boldsymbol{\mathcal{A}}\in\mathbb{R}^{n\times m}} : \boldsymbol{u}\cdot\boldsymbol{\mathcal{A}} = (\boldsymbol{u}^T\boldsymbol{\mathcal{A}})^T$
$\boldsymbol{:}$	:	double inner product	$\forall_{\boldsymbol{\mathcal{A}}\in\mathbb{R}^{m\times n},\ \boldsymbol{\mathcal{B}}\in\mathbb{R}^{n\times m}} : \boldsymbol{\mathcal{A}}\boldsymbol{:}\boldsymbol{\mathcal{B}} = \mathrm{tr}(\boldsymbol{\mathcal{A}}\boldsymbol{\mathcal{B}}) = \sum_i\sum_j A_{ij}B_{ji}$

Differential Operators

Symbol		Description	Mathematical Definition
$\nabla\cdot$	:	divergence operator	$\forall_{\boldsymbol{\mathcal{A}}\in\mathbb{R}^{n\times m}} : \nabla\cdot\boldsymbol{\mathcal{A}} = \left(\partial_{x_1},\ldots,\partial_{x_n}\right)^T\cdot\boldsymbol{\mathcal{A}}$
∇	:	gradient operator	$\forall_{\boldsymbol{u}\in\mathbb{R}^m} : \nabla\boldsymbol{u} = \left(\partial_{x_1}\boldsymbol{u},\ldots,\partial_{x_n}\boldsymbol{u}\right)^T$
Δ	:	Laplace differential operator	$\forall_{\boldsymbol{u}\in\mathbb{R}^m} : \nabla\cdot\nabla\boldsymbol{u}$

Dimensionless Variables

Symbol		Description	Mathematical Definition
Scalars			
p	:	pressure	
μ	:	viscosity	
Vectors			
$\boldsymbol{x}$	:	position	
$\boldsymbol{u}$	:	flow velocity	
Tensors			
$\mathcal{T}$	:	stress tensor	$\mathcal{T} = -p\mathcal{I} + \mu(\nabla_s \boldsymbol{u})_d$

Numerical Representations

Symbol		Description	Mathematical Definition
$\mathcal{K}$	:	coefficient matrix	
$\boldsymbol{q}$	:	load vector	
$\underset{\sim}{\hat{\alpha}}_k$	:	iterative solution after k^{th} iteration	
Δt	:	time step	
$\boldsymbol{r}_k$	:	iterative residual after k^{th} iteration	$\boldsymbol{r}_k = \mathcal{K}\underset{\sim}{\hat{\alpha}}_k - \boldsymbol{q}$
λ_k	:	k^{th} eigenvalue	
κ	:	spectral condition number	$\kappa = \lambda_{\text{max}}/\lambda_{\text{min}}$
ϵ	:	tolerance of stop criterium	
h	:	largest edge length in finite element mesh	
n_e	:	number of elements in finite element mesh	

Numbers

Symbol		Description
m	:	size of the coefficient matrix
n	:	space dimension

Physical Quantities

Symbol		Description	Unit	Mathematical Definition
Scalars				
t	:	time	[s]	
p	:	pressure	[Pa]	
ρ	:	density	[kg m^{-3}]	
μ	:	(dynamic) viscosity	[kg m^{-1}s^{-1}]	
c_p	:	specific heat	[J kg^{-1}K^{-1}]	
λ	:	effective conductivity	[W m^{-1}K^{-1}]	$\lambda = \lambda_c + \lambda_r$
λ_c	:	thermal conductivity	[W m^{-1}K^{-1}]	
λ_r	:	radiative conductivity	[W m^{-1}K^{-1}]	
Vectors				
$\boldsymbol{x}$	:	position	[m]	
$\boldsymbol{u}$	:	flow velocity	[m s^{-1}]	
$\boldsymbol{g}$	:	gravitational acceleration	[m s^{-2}]	
Tensors				
$\boldsymbol{\mathcal{T}}$	:	stress tensor	[N m^{-2}]	$\boldsymbol{\mathcal{T}} = -p\boldsymbol{\mathcal{I}} + \mu(\nabla_s \boldsymbol{u})_d$

Subscripts

Symbol		Description	Mathematical Definition
d	:	deviatoric part	$\forall_{\boldsymbol{\mathcal{A}} \in \mathbb{R}^{n\times n}} : (\boldsymbol{\mathcal{A}})_d = \boldsymbol{\mathcal{A}} - \frac{1}{3}\mathrm{tr}(\boldsymbol{\mathcal{A}})$

Unitary Mappings

Symbol		Description
$\boldsymbol{\mathcal{I}}_n$	:	$n \times n$ unity matrix
$\boldsymbol{e}_k$	:	k^{th} unity basis vector of $\mathbb{R}^n$, i. e. k^{th} column of $\boldsymbol{\mathcal{I}}_n$, $1 \leq k \leq n$

Contents

Chapter 1

Introduction

Nowadays, the presence of glass in our daily environment is simply indispensable. The utility of windows, bottles, drinking glasses, lenses, television screens and many other applications of glass strongly influences our present way of living. We might just look around us and ask ourselves: can we still live our normal lives without glass?

Although the range of applications of glass has rapidly increased in the last few centuries, the production of glass by melting raw materials is a process that humans have invented thousands of years ago. The earliest glass objects used by men were found in nature. These glass objects involved chipped pieces of obsidian, a natural volcanic glass, which were used by cave-dwellers for the production of primitive tools and weapons [34]. Not earlier than 7000 B.C., the Egyptians started making their own glass objects in the form of glass beads and jewelry. Still, it took until at least 1500 B.C., before the first glass containers were produced. The first bottles were produced by winding pieces of glass around moulds of concentrated sand and scraping the inside of the bottle [34]. A revolutionary course occurred when the Syrian craftsmen discovered the glass blowing pipe around 200 B.C. [20]. The invention of glass blowing eventually resulted in an improvement of glass jars and bottles and the production of glass drinking vessels by the Romans. Furthermore, the discovery of glass blowing in combination with colourants led to the invention of stained glass windows [34]. Currently, the glass manufacturing industry has reached the point, where copper wires can be replaced by glass optical fibers, which significantly reduces the amount of flaws in the transmission of information by telecommunication [34].

1.1 Glass Manufacturing

Before glass can be used for a specific purpose, a glass product has to be manufactured. This section describes some relevant glass manufacturing processes. Because this thesis is mainly concerned with glass formation, available glass forming methods are discussed into more detail. Relevant sources for glass manufacturing processes are [34] and [16]. Here, J. E. Shelby [34] is primarily concerned with the chemistry of glass forming, whereas D. Krause and H. Loch [16] focuss on the mathematics behind glass forming techniques.

Below some general glass manufacturing processes are described in the order of their application. The main source for these descriptions is [34].

Melting: In industry, the vast majority of glass products is manufactured by melting raw materials and recycled glass in tank furnaces at an elevated temperature [34, 16]. Examples of raw materials include silica, boric oxide, phosphoric oxide, soda and lead oxide. The temperature of the molten glass in the furnace usually ranges between 1200 and 1600 °C. A slow formation of the liquid is required to avoid bubble forming [34].

Forming: The glass melt is cut into uniform gobs, which are gathered in a forming machine. In the forming machine, the glass gobs slightly cool down to below 1200 °C. Thereafter, each individual molten gob is forced into the desired shape. In this stage, different types of products require different forming techniques. Available forming techniques are pressing, press-blowing, and blow-blowing, which are discussed further on. After the formation, the glass objects are rapidly cooled down as to take a solid form.

Annealing: Development of stresses during the formation of glass may lead to static fatigue of the product, or even to dimensional changes due to relaxation or optical refraction. The process of reduction and removal of stresses due to relaxation is called annealing [34].

In an annealing process, the glass objects are positioned in a so-called Annealing Lehr, where they are reheated to a uniform temperature region, and again gradually cooled down. The rate of cooling is determined by the allowable final permanent stresses and property variations throughout the glass [34].

Surface treatment: An exterior surface treatment is applied to reduce surface defects. Flaws in the glass surface are removed by chemical etching or polishing. Thereafter, flaw formation may be prevented by applying a lubricating coating to the glass surface. Crack growth is prevented by chemical tempering (ion exchange strengthening), thermal tempering or formation of a compressive coating. For more information about flaw removal and strengthening of the glass surface, the reader is referred to [34].

In this thesis, the principal process step is the actual glass formation. Therefore, special attention is paid to glass forming processes. In essence, there are three different forming techniques.

Pressing: Commercial glass pressing is a continuous process, where thin products (e. g. lenses, TV screens) are manufactured by pressing a gob that comes directly from the melt [34]. Initially, the gob is positioned in the centre of a mould. Over the mould, a plunger is situated. In order to enclose the space between the mould and the plunger, so that the glass cannot flow out, an additional ring is positioned on top of the mould border. During the pressing process, the plunger moves down through the ring and presses the gob into the desired shape. A picture from an industrial pressing process can be seen in Figure 1.1.

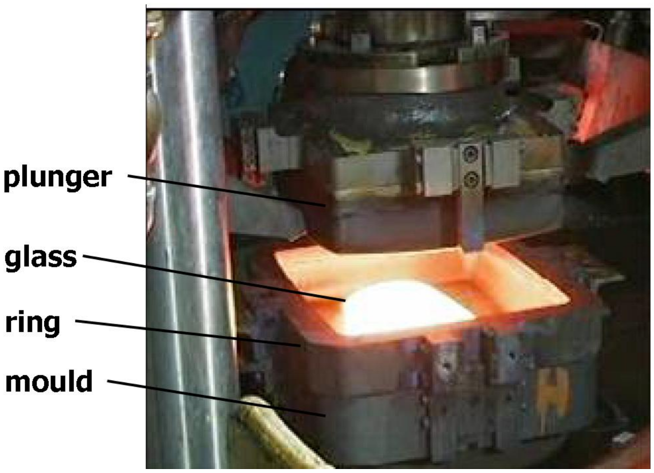


Figure 1.1: Picture of pressing process

Press-blowing: A hollow glass object is formed by inflating a preform with pressurised air. This is called the blowing stage of a press-blowing process. The preform is constructed by a pressing stage (see Figure 1.2). In a pressing stage, the gob enters a mould from above. Once the gob is inside the mould, the upper part of the mould is closed and the gob is pressed from below by a plunger. After the pressing stage, the resulting preform is carried to another mould for the blowing stage (see Figure 1.2). In the blowing stage, the glass is blown onto the mould wall. Here, a correct preform is important for an appropriate distribution of the glass over the mould wall.

Blow-blowing: The principle is the same as for press-blowing, except that the preform is produced by a blowing stage. This forming process is of minor importance for this project.

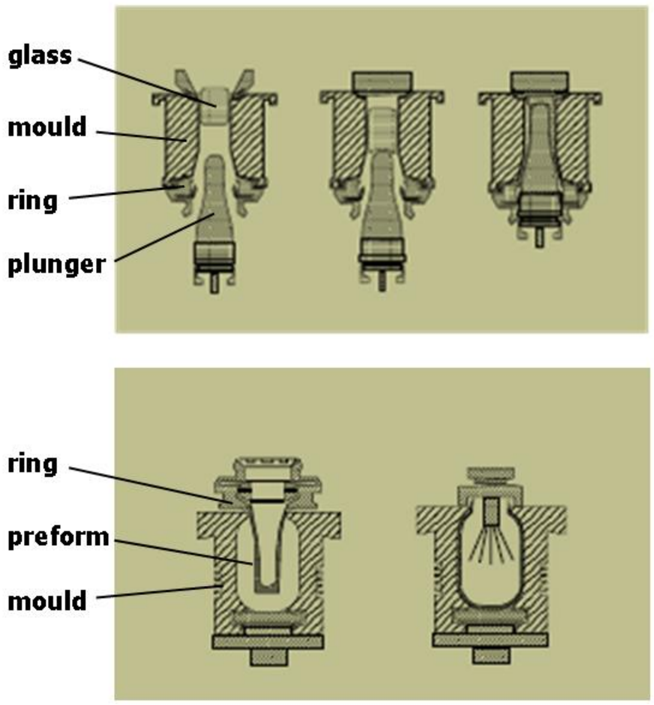


Figure 1.2: Schematic drawing of press-blowing process

1.2 Process Simulation

In industry, glass forming processes take place at high production rates. At the same time, glass manufacturers wish to optimise glass products to their customer's satisfactory. Unfortunately, practical experiments are in general quite expensive, whereas the majority has to be performed under complicated circumstances, e.g. high temperatures. In addition, the glass industry is fairly secretive about experimental data. Therefore, it is no surprise that in the recent past glass blowing techniques were based on experience, rather than on scientific research. To this day, computer simulation models have been necessary to gain improvement and a better understanding of glass blowing processes.

There are several reasons why glass manufacturers, material and equipment suppliers or third parties should be interested in process simulation. The main reasons for simulation of glass forming processes are the following.

Process analysis: to fully comprehend what exactly is taking place during glass formation. If both simulation results and results of practical experiments are available, they can be evaluated by comparing them with each other.

Process optimisation: to optimise an existing glass forming process with respect to speed, smoothness, strength, weight, cooling conditions, etc. Speculative methods to improve a glass forming process are often expensive and time consuming, whereas better results may be obtained by a thorough study of the physics behind the process. The effect of different settings (material, geometry, modelling) on a process can relatively easily be analysed by running several simulations. In this way, more optimal settings for the process can be obtained.

Process innovation: to analyse a completely new process. Before setting up a new forming process, it can already be studied and optimised by means of process simulation.

In order to analyse, optimise and innovate glass manufacturing process steps, TNO Glass Group (and some of its customers) has developed several 3D simulation tools. For simulation of glass furnaces, the finite volume simulation tool GTM-X is available, whereas for glass forming process steps, simulation tools based on the software library Sepran are being set up and maintained. This thesis considers two of these glass forming process simulation models: the blowing model for the blowing stage in press-blowing processes and the pressing model for pressing processes.

Sepran is a finite element software library[1] written in Fortran 77 language. It has originally been developed at Delft University of Technology and is currently maintained and distributed by the engineering firm Sepra. For an introduction to Sepran, the reader is referred to [29].

The glass forming process simulation models that are covered by this thesis revolve around three coupled, principal physical problems. Both the pressing process and the blowing process can be

[1] A software library that makes use of Finite Element Methods. Finite Element Methods are extensively explained in Chapter 4.

described by these problems, although the specific formulations will be different. They determine the physical behaviour of glass as well as air.

Flow problem: The motion of glass melt and air is described by the Navier-Stokes equations. For glass, the viscous forces dominate, so that the flow problem for glass can be reduced to a Stokes flow problem. The unknowns in the Stokes flow problem are the pressure and the flow velocity. The latter is used as convection velocity in the energy problem and the interface problem, which are described below

Energy problem: The energy exchange in glass, air and equipment is described by the heat equation. Since the viscosity of glass depends on the temperature and heat convection depends on the flow velocity, the energy problem is coupled to the flow problem.

Interface problem: The location of the glass-air interfaces is described by a convection problem. The solution of the convection problem is a function that determines the location of the glass-air interfaces. The material parameters in the flow problem and the energy problem depend on the location of the interface. Moreover, the convection problem depends on the flow velocity.

The problem descriptions for the pressing model are given in section 3.5. The blowing model is not discussed into detail in this thesis. For further details on the blowing model, the reader is referred to [13].

1.3 Problem Description

The boundary value problem in glass forming process simulation models are discretised by means of finite element methods and a suitable time discretisation scheme. Finally, an iterative solver is applied to solve the resulting system of equations for each successive time step.

Unfortunately, for the discrete Stokes flow problem, problems regarding calculation speed occur, as application of mesh refinement produces an excessive increase in the number of iterations. For TNO Glass Group's two dimensional pressing model, this increase is shown in Figure 1.3. From these figures, it can be observed that for a coarse mesh, the number of iterations only slightly increases as the mesh is refined. However for finer meshes, a substantial increase in the number of iterations is perceived for a relatively insignificant mesh refinement. Further mesh refinement even leads to stagnation of convergence or divergence of the solver. It should be mentioned that Sepran returns a substantial condition number of the order of magnitude $O(10^{22})$, even for the coarsest mesh considered in Figure 1.3.

It can be observed that this problem does not occur, or at least to a lesser degree, for the energy problem and the level set problem. In Figure 1.3, the contribution of the energy problem and the interface problem is coloured red on top of the bars for the number of accumulative iterations at

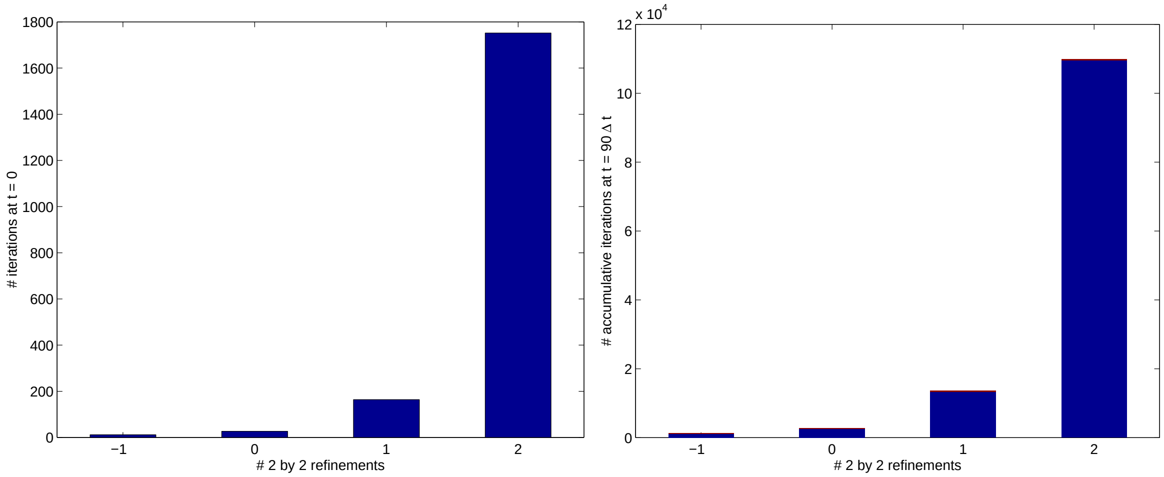


Figure 1.3: Increase in accumulative number of iterations for the BiCGstab solver with ILU preconditioning applied to the discrete Stokes flow problem in the pressing model. An absolute stop criterium is used, with tolerance ϵ = 1E-3. The refinements are 2^k over both dimensions with respect to the original mesh. Left: The number of iterations in the initial time step. Right: The number of iterations accumulated over all problems over all time steps up to $t = 90\Delta t$, with $\Delta t = \frac{1}{900}$. The contribution of the energy problem and the interface problem is coloured red on top of the bars.

$t = 90\Delta t$. Since this contribution can even hardly be noticed in the bargraph, it may reasonably be neglected. However, it is still important to consider the energy problem and the level set problem in the analysis of the solver performance, since the convergence of the solver for the discretised flow problem may be affected by the other problems, as the flow problem depends on the temperature and the location of the glass-air interfaces.

Although the problem is observed for the pressing model as well as the blowing model, a full problem analysis is considered for the pressing model only. This consideration is not fundamentally restrictive, since similar physical problems are solved, and also the corresponding iterative solvers show the same behaviour for both models. There are two main reasons to prefer the model pressing over the blowing model. Firstly, it is interesting to include the effect of mesh deformation in the pressing model. Secondly, the pressing model has been developed further than the blowing model. A disadvantage is that the source code for TNO Glass Group's pressing model is extensive and subdivided into a lot of subordinate Fortran files, which makes it considerably more time consuming to study the pressing model. Where appropriate, results for the pressing model can also be used in the blowing model.

1.4 Objectives

The main objective is to analyse the solver performance for the flow problem in the glass forming process simulation models, with improvement of calculation speed and robustness as motivation. In particular, it is analysed whether the linear solvers do not require an unnecessary large number of iterations for a reasonably accurate solution. In addition, the applied modelling is studied with respect to the final linear system of equations. Improvement of the solver performance may be achieved by changing numerical settings, applying different mathematical methods or by disregarding any physical phenomena in the existing model.

1.5 Thesis Outline

This thesis is structured as follows. First, Chapter 2 focusses on the physical aspects of glass pressing. Then, Chapter 3 uses the physics to derive a mathematical model for general pressing processes. At the end of the chapter, some particular mathematical aspects of TNO Glass Group's pressing model are discussed. Chapter 4 explains finite element methods for general convection diffusion problems. The finite element discretisation of the energy problem and the interface problem are covered by this theory. Chapter 5 goes into further detail onto finite element methods for Stokes flow problems. In this chapter, a literature review of different discretisation and solution methods is given and the methods used in TNO Glass Group's pressing model are critically examined. In addition, Chapter 6 discusses the performance of iterative solvers, as well as preconditioners and linear solvers for Stokes flow problems. Some particular attention is paid to ILU factorisation preconditioning, which is used in TNO Glass Group's pressing model. Subsequently, Chapter 7 examines numerical results of some test cases, both for TNO Glass Group's axi-symmetric pressing model and a simple test model for the simulation of a pressing process time step. Furthermore, numerical solutions are visualised and analysed. Finally, Chapter 8 states the conclusions and gives some recommendations.

Chapter 2

Physical Aspects of Glass Pressing

Before describing glass pressing models, it is convenient to study the physics of glass pressing processes. This chapter deals with relevant physical aspects of glass pressing, which are subdivided into three separable sections. Section 2.1 looks at the physics of glass pressing processes, section 2.2 deals with the material properties of glass and how they relate to the glass temperature, and section 2.3 is concerned with heat transfer in glass. For most parts of this chapter, the reader is referred to [38].

2.1 Glass Pressing Processes

In a glass pressing process, a glass gob is placed in the centre of a mould and pressed from above by a plunger (see Figure 2.1). The mould-plunger construction is closed from above by fixing a ring that encloses the bottom surface of the plunger on top of the mould border.

The temperature of the mould-plunger construction is typically about 500 °C. Because of the high temperature of the glass gob (typically about 1000 °C), the surface temperature of the material will increase. To keep the temperature of the material within acceptable bounds, the mould and plunger are heat insulated by means of water-cooled channels.

The pressing process is initiated by applying an external force F_e to the plunger. This causes the plunger to move down with velocity $V_p(t)$. This plunger velocity is the result of the total force F on the plunger, which is the sum of the external force F_e and the force of the glass on the plunger F_g:

$$\frac{\mathrm{d}V_p}{\mathrm{d}t} = \frac{F}{m_p} = \frac{F_e + F_g}{m_p}, \tag{2.1.1}$$

where m_p is the mass of the plunger. The force of the glass on the plunger is caused by the plunger movement itself. Clearly, if a constant external force is applied to the plunger, the plunger moves down until the force of the glass on the plunger is equal in magnitude to the external force. S. W. Rienstra and T. D. Chandra [27] and K. Laevsky [17] deduce that the force F_g can be expressed as a linear

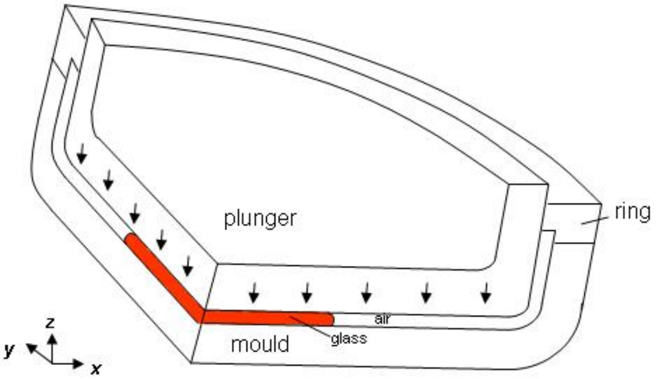


Figure 2.1: Set-up for an axi-symmetric glass pressing process.

function of the plunger velocity V_p. So, in case of a constant external force, the plunger velocity decreases exponentially in time.

2.2 Material Properties of Glass

In glass forming processes, the viscosity of glass plays an important role. The viscosity measures the resistance to shear of a medium. The range of viscosity at relatively low temperatures is huge for glass: it amounts from 10 Pa s at the melting temperature (about 1500 °C) to 10^{20} Pa s at room temperature. The viscosity increases continuously as a glass melt is cooled, which makes glass forming possible. In appropriate glass forming processes, the viscosity becomes so high just after the glass is formed, that the glass melt retains its shape. Subsequently, the fixed glass form is cooled to a solid state. Typical values for the viscosity in glass forming processes lie between 10^3 and 10^4 Pa s [38].

An important characteristic glass temperature is the transformation temperature T_g. The transformation temperature is the temperature, for which the transition from properties corresponding to liquid glass to properties corresponding to solid glass occurs. Typical values of the viscosity at the transition temperature lie between 10^{11} and 10^{12} Pa s. The temperature dependence for the viscosity of glass above the transformation temperature T_g is given by the VFT-relation, due to Vogel, Fulcher and Tamman [38]:

$$\mu(T) = 10^{-A+B/(T-T_L)}. \tag{2.2.1}$$

Here, A [log(Pa s)], B [°C log(Pa s)] and T_L [°C] are the Lakatos coefficients, which depend on the composition of the glass melt. Below T_g, the glass structure depends on the cooling speed and no obvious viscosity-temperature relation is available [38].

For high temperatures, glass behaves as a Newtonian fluid. In this case, the glass melt is isotropic and the stress tensor satisfies [38]

$$\mathcal{T} = -p\mathcal{I} + 2\mu(\nabla_s \boldsymbol{u})_d. \tag{2.2.2}$$

The density-temperature relation is

$$\rho(T) = \rho_0\big(1 - \beta(T - T_0)\big), \tag{2.2.3}$$

where

- β [$°\mathrm{C}^{-1}$] is the volumetric expansion coefficient,
- T_0 [°C] is a reference temperature,
- ρ_0 [kg m^{-3}] is the density at the reference temperature.

The volumetric expansion coefficient is often assumed constant and is above the transformation temperature T_g typically ranged from $5 \cdot 10^{-5}$ to $8 \cdot 10^{-5}$ $°\mathrm{C}^{-1}$. This makes it in general quite reasonable to assume incompressibility. The density of glass above T_g is of the order 2300 to 2500 kg m^{-3} and is 5 to 8% lower than at room temperature [38].

2.3 Heat Transfer in Glass

For glass, a global classification of the types of heat transfer dominating can be made:

to 300 °C	:	conduction,
from 300 to 800 °C	:	conduction and radiation,
from 800 °C	:	radiation and convection.

The contribution of both conduction and radiation results in a heat flux [W m^{-2}]

$$\boldsymbol{q} \;=\; -\lambda \cdot \nabla T, \tag{2.3.1}$$

where λ is the effective conductivity [W $\mathrm{m}^{-1}\mathrm{K}^{-1}$], given by

$$\lambda = \lambda_c + \lambda_r. \tag{2.3.2}$$

Here, λ_c is the thermal conductivity and λ_r is the radiative conductivity. The heat conduction coefficient measures 1.0 W $m^{-1}K^{-1}$ at room temperature for NaCa-glass and increases with approximately 0.1 W $m^{-1}K^{-1}$ per 100 K. Calculation of the radiative conductivity is often a complicated process. However, for non-transparant glasses, the radiative heat conductivity λ_r can be simplified by the Rosseland approximation

$$\lambda_r(T) = \frac{16}{3}\frac{n^2\sigma T^3}{\alpha}, \tag{2.3.3}$$

where

- σ is the Stefan Boltzmann radiation constant [W $m^{-2}K^{-4}$],
- n is the average refractive index [-],
- α is the absorption coefficient [m^{-1}].

The radiative conductivity λ_r in the sense of (2.3.3) is called the Rosseland parameter. Clearly, the Rosseland parameter strongly depends on the temperature. This relation cannot be applied for highly transparent glasses, since in this case not all radiation is absorbed by the glass melt. For more information on heat transfer in glass, the reader is referred to [20, 38]. The NCNG Handbook for Glass Fabrication by H. de Waal and R. G. C. Beerkens [38] has been used as a source for this section.

Chapter 3

Mathematical Pressing Model

This chapter presents a mathematical model for glass pressing. As mentioned in section 1.2, glass pressing processes are described by three coupled physical problems, which are explained in three different sections of this chapter:

1. a flow problem for the motion of glass and air (section 3.2),
2. an energy problem for the energy exchange between glass, air and equipment (section 3.3),
3. an interface problem for the location of the glass-air interface (section 3.4).

For each of these problems a mathematical boundary value problem (BVP) can be set up. Together, the three BVPs form a mathematical model for glass pressing.

This chapter is structured as follows. First section 3.1 defines the physical domains into which a glass pressing construction can be subdivided and in which the BVPs are defined. Then, section 3.2-3.4 set up and analyse the BVPs. Finally, section 3.5 discusses the mathematical model in TNO Glass Group's pressing model.

3.1 Geometry

In order to formulate a mathematical model, the axi-symmetric glass pressing construction is subdivided into separate subdomains. First, separate domains for the equipment (mould, plunger and ring) are considered. The mould and plunger can again consist of parts with different material properties, for which separate subdomains should be defined. These subdomains of the equipment are only of interest for the energy exchange problem and are therefore not discussed into detail in this thesis. In the centre of the glass pressing construction, between the mould and the plunger, an axi-symmetric glass gob is located. The remaining space between the mould, plunger and ring is filled with air. The

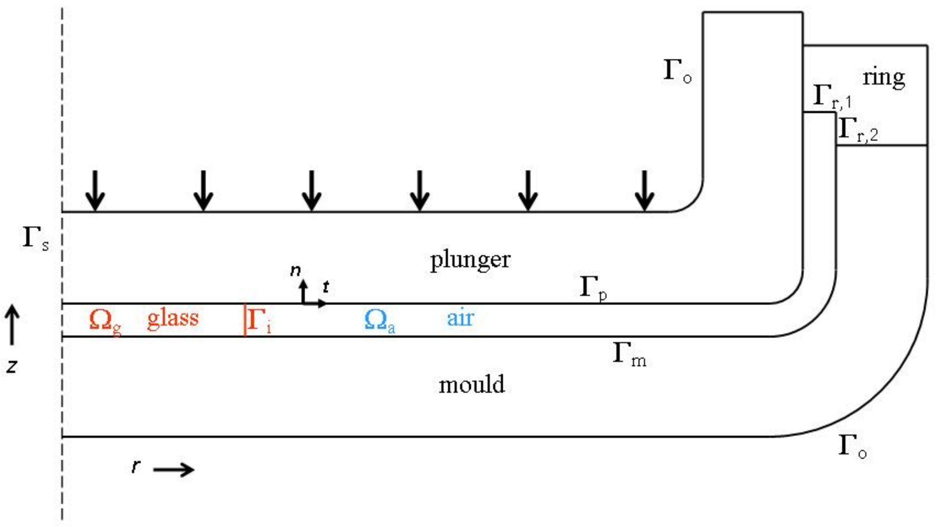


Figure 3.1: Problem domains in pressing model

glass domain and the air domain are relevant for all physical problems involved. Their dimensions change in time as the plunger moves down.

Figure 3.1 illustrates the subdivision of the glass pressing construction into subdomains by means of a 2D-cut from the symmetry axis. The entire domain for the glass pressing construction, consisting of equipment, glass melt and air, is denoted by Σ. Domain Σ is enclosed by $\Gamma_o \cup \Gamma_s$, where Γ_o is the outer boundary of the equipment and Γ_s is the axis of symmetry of the entire domain. Let Ω_g and Ω_a be the glass domain and the air domain, respectively, separated by an interface Γ_i, as illustrated in Figure 3.1. The boundary $\partial\Omega$ of $\Omega := \Omega_g \cup \Omega_a$ consists of four parts:

$$\partial\Omega = \Omega \cap (\Gamma_s \cup \Gamma_p \cup \Gamma_r \cup \Gamma_m), \tag{3.1.1}$$

where the indices $s,\ p,\ r,\ m$ denote symmetric, plunger, ring and mould, respectively. Here,

$$\Gamma_r = \Gamma_{r,1} \cup \Gamma_{r,2}, \tag{3.1.2}$$

where $\Gamma_{r,1}$ and $\Gamma_{r,2}$ are separated by the inner corner of the ring. Furthermore on $\partial\Omega$, an outer normal $\boldsymbol{n}$ and a tangential $\boldsymbol{t}$ are given.

3.2 Stokes Flow Problem

Equations

The motion of glass melt and air is described by the Navier-Stokes equations for incompressible fluids. These involve the momentum equations,

$$\rho\Big(\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u}\cdot\nabla\boldsymbol{u}\Big) = -\nabla p + \rho\boldsymbol{g} + \nabla\cdot(2\mu\nabla_s\boldsymbol{u})_d, \qquad \text{in}\,\Omega, \tag{3.2.1}$$

and the continuity equation,

$$\nabla\cdot\boldsymbol{u} = 0, \qquad \text{in}\,\Omega. \tag{3.2.2}$$

Here, the unknowns are the flow velocity $\boldsymbol{u}$ [m s^{-1}] and the pressure p [Pa]. Note that the material properties depend on the location of the glass-air interface. For glass the viscosity is temperature dependent (see section 2.2), whereas for air the viscosity can be assumed to be uniform. Typical values for the Navier-Stokes flow problem are:

glass density	:	$\rho_{0,g}$	=	$2.5\cdot10^3$ kg m^{-3},
glass viscosity	:	$\mu_{0,g}$	=	10^4 kg m^{-1}s^{-1},
air density	:	$\rho_{0,a}$	=	1.0 kg m^{-3},
air viscosity	:	$\mu_{0,a}$	=	10^{-5} kg m^{-1}s^{-1},
gravitational acceleration	:	g_0	=	9.8 m s^{-2},
pressing time	:	τ	=	1.5 s,
length scale of the glass gob	:	L	=	10^{-2} m.

Furthermore from L and τ, a typical velocity can be derived

$$V = \frac{L}{\tau}. \tag{3.2.3}$$

In order to apply a quantitative analysis, the Navier-Stokes equations are written in dimensionless form. Introduce dimensionless variables

$$t^* := \frac{t}{\tau}, \qquad \boldsymbol{x}^* := \frac{\boldsymbol{x}}{L}, \qquad \boldsymbol{u}^* := \frac{\boldsymbol{u}}{V}, \qquad p^* := \frac{Lp}{\mu_0 V}. \tag{3.2.4}$$

In addition, define the dimensionless viscosity and gravitational acceleration:

$$\mu^* = \frac{\mu}{\mu_0}, \qquad \boldsymbol{g}^* = \frac{\boldsymbol{g}}{g_0}. \tag{3.2.5}$$

For convenience, all dimensionless variables, spaces and operators with respect to the dimensionless variables are denoted with superscript *. Substitution of the dimensionless variables into the

Navier-Stokes equations and division by the order of magnitude of the diffusion term, $\frac{\mu_0 V}{L^2}$, leads to the dimensionless Navier-Stokes equations

$$\begin{aligned}\mathrm{Re}\Big(\frac{\partial \boldsymbol{u}^*}{\partial \mathrm{t}^*} + \boldsymbol{u}^* \cdot \nabla^* \boldsymbol{u}^*\Big) = -\nabla^* p^* + \frac{\mathrm{Re}}{\mathrm{Fr}} \boldsymbol{g}^* + \nabla^* \cdot (2\mu^* \nabla^*_s \boldsymbol{u}^*)_d, &\quad \text{in}\, \Omega^*, \qquad (3.2.6)\\ \nabla^* \cdot \boldsymbol{u}^* = 0, &\quad \text{in}\, \Omega^*, \qquad (3.2.7)\end{aligned}$$

where

$$\begin{aligned}\mathrm{Re} &= \frac{\rho_0 V L}{\mu_0}, \qquad (3.2.8)\\ \mathrm{Fr} &= \frac{V^2}{g_0 L} \qquad (3.2.9)\end{aligned}$$

are the Reynolds number and the Froude number, respectively. Typical values for the dimensionless numbers are

$$\mathrm{Re_g} = 1.7 \cdot 10^{-5}, \qquad \mathrm{Re_a} = 6.7, \qquad \mathrm{Fr} = 4.5 \cdot 10^{-3} \qquad (3.2.10)$$

From the small Reynolds number for glass, it can be concluded that the inertia forces can be neglected with respect to the viscous forces. Furthermore,

$$\frac{\mathrm{Re_g}}{\mathrm{Fr}} = 3.8 \cdot 10^{-3},$$

which means that also the contribution of gravitational forces is rather small. On the other hand, air is hardly viscous, so that the flow of air is prescribed by the full Navier-Stokes equations. Since the specific transport phenomena of air are not of any interest in the pressing model, the air in the mathematical model is replaced by a fictive fluid to simplify the transport calculations. This fictive fluid has viscosity 10 Pa, whereas its other material properties are the same as for air. This viscosity is still much smaller than for glass, so that physical phenomena at the transition from glass to fictive fluid are properly described. The Reynolds number for the fictive fluid is

$$\mathrm{Re_f} = 6.7 \cdot 10^{-6}. \qquad (3.2.11)$$

Hence the contribution of inertia forces and gravitational forces is negligible. Thus for both glass and fictive fluid, the Stokes equations remain:

$$\nabla^* \cdot \mathcal{T}^* = \mathbf{0}, \qquad \nabla^* \cdot \boldsymbol{u}^* = 0, \qquad \text{in}\, \Omega^*, \qquad (3.2.12)$$

where $\mathcal{T}^*$ is the dimensionless stress tensor, which satisfies (2.2.2) in terms of the dimensionless variables.

Boundary Conditions

The flow BCs are obtained as follows. On Γ_s, symmetry is assumed. For glass, no-slip BCs are adopted, whereas for the fictive fluid, free-slip BCs are proposed. This is reasonable as long as the viscosity of the fictive fluid is much smaller than the viscosity of glass. In order to simplify the flow calculations, let the fictive fluid flow freely through the upper part $\Gamma_{r,1}$ of the ring. In conclusion, the flow BCs can be formulated as

$$
\begin{array}{rcrclrcl}
\Gamma_s \cap \Omega_g & : & \boldsymbol{u}\cdot\boldsymbol{n} & = & 0, & \boldsymbol{\mathcal{T}}\boldsymbol{n}\cdot\boldsymbol{t} & = & 0, \\
(\Gamma_r \cup \Gamma_m) \cap \Omega_g & : & \boldsymbol{u}\cdot\boldsymbol{n} & = & 0, & \boldsymbol{u}\cdot\boldsymbol{t} & = & 0, \\
\Gamma_p \cap \Omega_g & : & (\boldsymbol{u}-\boldsymbol{u}_p)\cdot\boldsymbol{n} & = & 0, & (\boldsymbol{u}-\boldsymbol{u}_p)\cdot\boldsymbol{t} & = & 0, \\
\Gamma_m \cap \Omega_a & : & \boldsymbol{u}\cdot\boldsymbol{n} & = & 0, & \boldsymbol{\mathcal{T}}\boldsymbol{n}\cdot\boldsymbol{t} & = & 0, \\
\Gamma_{r,1} \cap \Omega_a & : & \boldsymbol{\mathcal{T}}\boldsymbol{n}\cdot\boldsymbol{n} & = & 0, & \boldsymbol{\mathcal{T}}\boldsymbol{n}\cdot\boldsymbol{t} & = & 0, \\
\Gamma_{r,2} \cap \Omega_a & : & \boldsymbol{u}\cdot\boldsymbol{n} & = & 0, & \boldsymbol{\mathcal{T}}\boldsymbol{n}\cdot\boldsymbol{t} & = & 0, \\
\Gamma_p \cap \Omega_a & : & (\boldsymbol{u}-\boldsymbol{u}_p)\cdot\boldsymbol{n} & = & 0, & \boldsymbol{\mathcal{T}}\boldsymbol{n}\cdot\boldsymbol{t} & = & 0,
\end{array}
$$

where, $\boldsymbol{u}_p = V_p\boldsymbol{e}_z$ is the plunger velocity. For more information on the modelling of the flow BCs, the reader is referred to [27].

3.3 Energy Exchange Problem

Equations

The energy exchange between glass, air (fictive fluid) and equipment is described by the heat equation for incompressible media:

$$
\rho c_p\left(\frac{\partial T}{\partial t} + \boldsymbol{u}\cdot\nabla T\right) = \nabla\cdot(\lambda\nabla T) + \mu(\nabla_s\boldsymbol{u} : \nabla\boldsymbol{u}), \qquad \text{in}\,\Sigma, \tag{3.3.1}
$$

where the temperature T [K] is unknown. The energy problem is coupled to the flow problem, since the viscosity is temperature dependent and the flow velocity appears in (3.3.1). In addition to the typical values for the flow problem in section 3.2, the following typical values for the energy exchange problem are considered:

initial glass temperature	:	T_1	=	1000 °C,
initial equipment temperature	:	T_0	=	500 °C,
specific heat of glass	:	$c_{p,g}$	=	$1.4\cdot10^3$ J kg^{-1}K^{-1},
specific heat of air	:	$c_{p,a}$	=	10^3 J kg^{-1}K^{-1},
thermal conductivity of glass	:	$\lambda_{c,g}$	=	5 W m^{-1}K^{-1},
thermal conductivity of air	:	$\lambda_{c,a}$	=	10^{-2} W m^{-1}K^{-1}.

In order to write the heat equation in dimensionless form, introduce the dimensionless temperature

$$T^* := \frac{T - T_0}{T_1 - T_0}. \tag{3.3.2}$$

Substituting the dimensionless variables into the heat equation and splitting up the effective conductivity into (2.3.2) give the dimensionless form

$$\mathrm{Pe}(\frac{\partial \mathrm{T}^*}{\partial \mathrm{t}^*} + \boldsymbol{u}^* \cdot \nabla^* \mathrm{T}^*) = \Delta^* \mathrm{T}^* + \nabla^* \cdot \Big(\frac{\lambda_{\mathrm{r}}}{\lambda_{\mathrm{c}}} \nabla^* \mathrm{T}^*\Big) + \mathrm{Br} \cdot \mu^* (\nabla_{\mathrm{s}}^* \boldsymbol{u}^* : \nabla^* \boldsymbol{u}^*), \qquad \text{in}\, \Sigma^*, \tag{3.3.3}$$

where

$$\mathrm{Pe} = \frac{\rho_0 c_p V L}{\lambda_c}, \tag{3.3.4}$$

$$\mathrm{Br} = \frac{\mu_0 V^2}{\lambda_c (T_1 - T_0)} \tag{3.3.5}$$

are the Péclet number and the Brinkman number, respectively. For the energy exchange problem, they have typical values

$$\mathrm{Pe_g} = 46.7, \qquad \mathrm{Pe_a} = 6.7, \qquad \mathrm{Br_g} = 1.8 \cdot 10^{-4}, \qquad \mathrm{Br_a} = 8.9 \cdot 10^{-11}. \tag{3.3.6}$$

The Péclet number represents the ratio between the contribution of diffusion and convection. Apparently, the heat transport in both glass and air is prescribed by thermal convection as well as diffusion. On the other hand, the Brinkman numbers have insignificant values, and hence the influence of viscous dissipation can be neglected. Thus, the heat equation simplifies to

$$\frac{\partial T^*}{\partial t^*} + \boldsymbol{u}^* \cdot \nabla^* T^* = \nabla^* \cdot (\lambda^* \nabla^* T^*), \qquad \text{in}\, \Sigma^*, \tag{3.3.7}$$

with

$$\lambda^* = \frac{1}{\mathrm{Pe}} \frac{\lambda}{\lambda_c}. \tag{3.3.8}$$

Boundary Conditions

The energy BCs follow from symmetry and heat exchange with the surroundings. This induces the BCs

$$\begin{array}{rcrcl} \Gamma_s & : & (\lambda \nabla T) \cdot \boldsymbol{n} & = & 0, \\ \Gamma_o & : & (\lambda \nabla T) \cdot \boldsymbol{n} & = & \alpha (T - T_\infty), \end{array}$$

where T_∞ is the temperature of the surroundings. Here, the heat transfer coefficient α can differ for the mould, the ring or the plunger. On interfaces between different media, a steady state temperature transition is imposed, that is, for a transition from medium 1 to medium 2,

$$\Big[\lambda \frac{\partial T}{\partial n}\Big]_1 = \Big[\lambda \frac{\partial T}{\partial n}\Big]_2.$$

3.4 Glass-Air Interface Problem

The location of the glass-air interfaces is described by a convection problem for a so-called level set function [33]:

$$\frac{\partial \theta}{\partial t} + \boldsymbol{u} \cdot \nabla \theta = 0. \tag{3.4.1}$$

Level set functions can be used to track the interface and label the subdomains:

$$\begin{aligned} &\theta < 0, && \text{in air (fictive fluid),} \\ &\theta = 0, && \text{on the glass-air interface,} \\ &\theta > 0, && \text{in glass.} \end{aligned}$$

The glass-air interface problem is coupled to the Stokes flow problem and the energy exchange problem, since the material properties depend on the location of the interface and the flow velocity appears in the convection term of (3.4.1).

3.5 Mathematical Aspects of TNO Glass Group's Pressing Model

Industrial modelling of glass pressing is generally recognised as a difficult problem. It involves a physical system of two-phase Stokes flow coupled to heat exchange in fluids with moving boundaries, which results in large-scale computations. At the moment, TNO Glass Group's glass pressing simulation model is one of the few process simulation models in the world that is able to simulate glass pressing processes within a reasonable amount of CPU time. Consequently, specific information on mathematical glass pressing process simulation models is limited.

Apart from the process simulation models for glass pressing processes, also process simulation models for the pressing stage in press-blowing processes have been constructed. In theory, these glass pressing model are quite similar. For more information on process simulation models for the glass bottle and jar pressing stage, the reader can be referred to K. Laevsky [17].

It is noteworthy that TNO Glass Group's glass pressing model has a few extensions upon the pressing model that is dealt with in this thesis. Most importantly, TNO Glass Group's glass pressing model can also simulate glass pressing processes for three dimensional, not necessarily axi-symmetric geometries. For convenience, in this report TNO Glass Group's pressing model is referred to as TNO Glass Group's axi-symmetric model if the model is specifically applied to an axi-symmetric problem, and as TNO Glass Group's three-dimensional model for a general three-dimensional problem. In fact, the solver problem originates from the three-dimensional pressing model, and is much more severe in this model than in the axi-symmetrical model. However, because this three dimensional glass pressing model takes up a lot of CPU time and the iterative solver performance is relatively instable, the axisymmetrical model is used to analyse the solver performance in this project. TNO Glass Group's glass pressing model also has other features, such as radiation calculations and an annealing process simulation model. Usually radiation calculations are switched off in the pressing model, since they are considerably expensive.

It is as well important to remark that TNO Glass Group's glass pressing model uses dimensional problems, whereas the order of magnitude of each term in the PDEs is assessed as in the dimensionless equations in Chapter 3. Although using dimensional problems cannot be recommended, it is shown in Chapter 7 that for glass pressing process simulation models the solver performance and numerical results are hardly affected if the problems are dimensional instead of dimensionless. However, in order to prove correctness of the pressing model, this thesis generally applies the dimensionless form of the problems.

The boundary value problems in TNO Glass Group's glass pressing model are discretised by means of finite elements methods (FEMs) and a suitable time discretisation scheme (e.g. Euler implicit or Crank Nicolson). The discretisation schemes have been implemented in the finite element package Sepran, which is briefly discussed in section 1.2. The FEMs applied are thoroughly explained in Chapter 4 and Chapter 5.

In order to switch between no-slip and free-slip BCs across the moving glass-air interface, mixed BCs are applied in TNO Glass Group's glass pressing model. In tangential direction, these read

$$(\mathcal{T}\boldsymbol{n} + \beta\boldsymbol{u}) \cdot \boldsymbol{t} = \beta V_s, \tag{3.5.1}$$

where β is the wall friction coefficient [N m^{-3}s] and V_s is the slip velocity [m s^{-1}]. Clearly, if $\mathcal{T}\boldsymbol{n} \cdot \boldsymbol{t} = 0$ is desired, then $\beta = 0$ suffices. On the other hand, if a restriction for the tangential velocity is imposed, then theoretically $\beta \to \infty$, which comes to introducing a sufficiently large numerical value for the wall friction coefficient. A disadvantage of this approach is that the large wall friction coefficient also appears in the stiffness matrix, causing an oversized condition number.

To solve the resulting system of equations for each successive time step an iterative solver for non-symmetric problems is used, usually BiCGstab [37, 28, 26] with an incomplete LU factorisation preconditioner [28, 2]. Other iterative solvers for non-symmetric problems available in Sepran are CGS, GMRES, GCR, GMRESR (a method with GMRES as inner loop and GCR as outer loop using a variable polynomial preconditioner) and overrelaxation. Other relevant preconditioners in Sepran are (modified) Eisenstat, Gauss Seidel and block SSOR [31]. Several iterative solvers and preconditioners are examined in Chapter 6. For more information on iterative solvers and preconditioners, the reader is referred to [28].

In Figure 3.2, a flow chart of TNO Glass Group's pressing model is given. Here, it should be mentioned that, although in this thesis the media are assumed to be incompressible, the density in the TNO Glass Group's pressing model can be taken temperature dependent (see section 2.2).

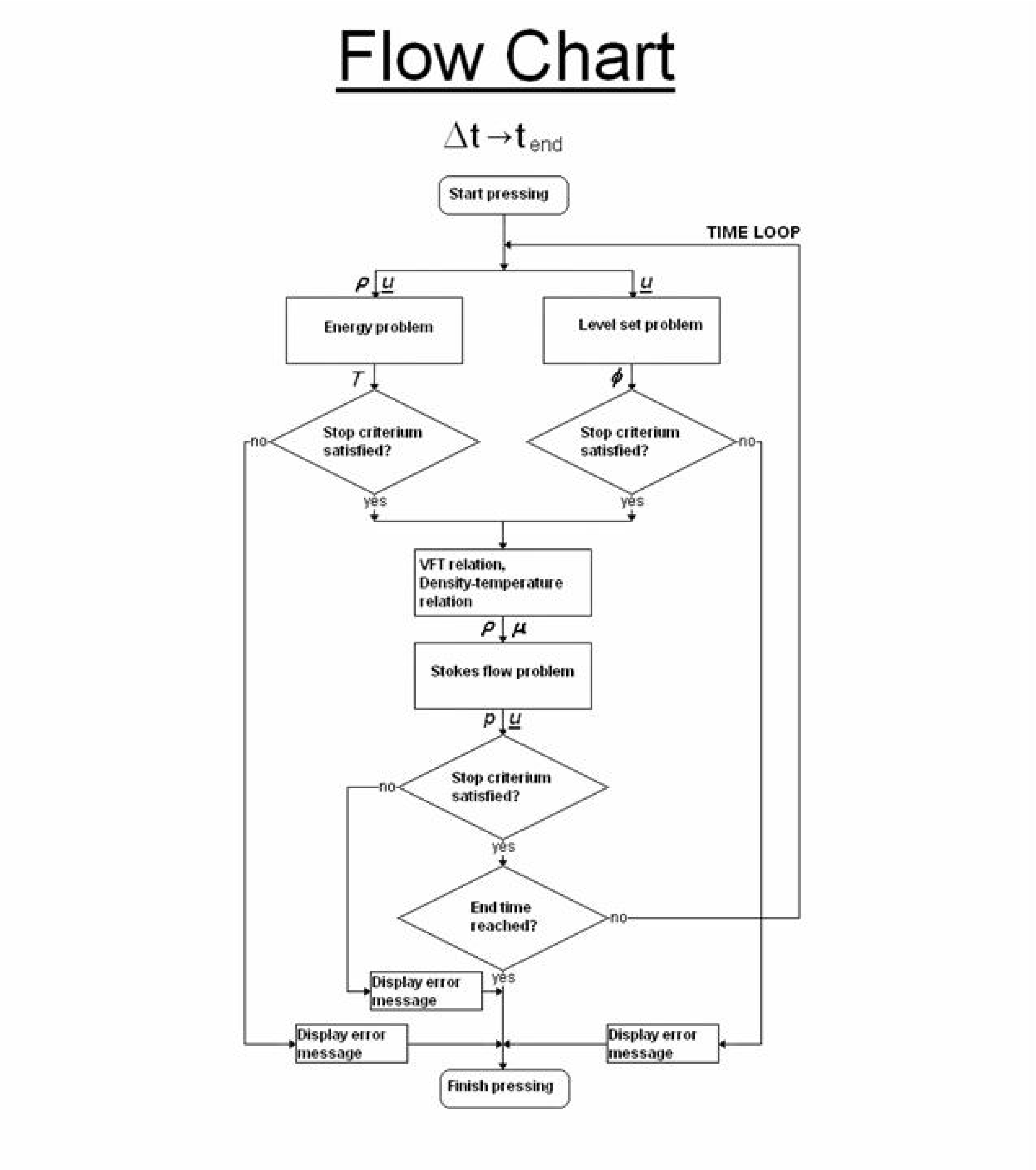


Figure 3.2: Flowchart of the pressing model. At time $t = 0$, only a flow problem is solved. These flow solutions are used at $t = \Delta t$.

Chapter 4

Galerkin Finite Element Methods

Galerkin finite element methods are essential building blocks of TNO Glass Group's glass forming process simulation models. Therefore, it is worth studying the basics of finite element methods (FEMs). The objective of this chapter is to offer sufficient background in FEMs to understand the principles of these glass forming process simulation models. The specific finite element theory for Stokes flow problems is studied in Chapter 5. The main sources of this chapter are [19], [8] and [3].

Galerkin FEMs approximate the solution of a partial differential equation by means of a linear combination of some suitable functions. These functions are defined with the internal structure of the elements that partition the domain of interest.

FEMs have some considerable advantages compared to other numerical methods. Notably, FEMs can relatively easily be applied on complex domains, since the domain can be partitioned into elements of a much simpler structure. Furthermore, computations can often be restricted to reference elements that represent the structure of groups of elements, which significantly simplifies calculations (J. M. L. Maubach, [19]).

This chapter is structured as follows. For the sake of illustration, Galerkin FEMs are applied to convection-diffusion problems. Section 4.1 introduces the general form of convection-diffusion equations, while section 4.2 states the boundary conditions (BCs). Subsequently, section 4.3 writes the convection-diffusion problem in a so-called weak form. The result is called a variational problem. Galerkin FEMs attempt to approximate the solution of the variational problem by a linear combination of so-called basis functions, which span finite dimensional subspaces of the vector spaces that are subject to the variational problem. Section 4.4 shows how such subspaces are constructed for a suitable set of basis functions. Then, section 4.5 applies Galerkin FEMs to the variational problem, which results in a discrete convection-diffusion problem. Section 4.6 discusses some suitable properties that basis functions should satisfy, so that the discrete problem can relatively easily be solved. It appears that a suitable construction of the basis functions is accompanied by a partition of the domain into so-called elements. Section 4.7 illustrates how the coefficients and right hand side of the discrete convection-diffusion problem can be assembled by adding the contributions of all elements. Next, section 4.8 deals with so-called isoparametric elements, for which the finite elements solutions are approximated in the same way as the geometry. For completeness, section 4.9 briefly discusses a numerical integration scheme that is widely used in FEMs. Finally, section 4.10 shows how numerical boundary conditions can be incorporated into the discrete problem.

4.1 Convection-Diffusion Equations

It is instructive to apply the theory of FEMs in this section to convection-diffusion equations. Both the heat equation and the convection equation for the glass-air interface in the glass forming process simulation models are convection-diffusion equations. Moreover, FEMs for convection-diffusion equations can easily be extended to more general types of partial differential equations (PDEs), including Stokes flow equations, as can be seen in Chapter 5.

Convection diffusion equations have the following general form. Let $n \in \mathbb{N}$ be the dimension of the problem. Consider a region of interest $\Omega \subset \mathbb{R}^n$ that is open and polygonal bounded. Find $u \in C^2(\Omega)$ such that

$$L(u) = f \qquad \text{in}\, \Omega, \tag{4.1.1}$$

where L is the convection-diffusion differential operator, defined by

$$L(u) := -\nabla \cdot (d\nabla u) + \boldsymbol{c} \cdot \nabla u + s\, u. \tag{4.1.2}$$

The coefficients in (4.1.2) are known as:

- the diffusion function $d \in L^\infty(\Omega)$,
- the convection vector $\boldsymbol{c} \in \left\{\gamma \in L^\infty(\Omega) \,\middle|\, \|\nabla\gamma\|_\infty < \infty\right\}^n$,
- the source function $s \in L^\infty(\Omega)$,
- the right-hand side function $f \in L^2(\Omega)$.

For definitions of the function spaces, the reader is referred to appendix A.

4.2 Boundary Conditions

In order to find a unique solution of (4.1.1)-(4.1.2), BCs are required. Let $\partial\Omega$ denote the boundary of Ω. The boundary can be split into several parts, on which one of the following three BCs holds:

1. Dirichlet BC on $\Gamma_D \subset \partial\Omega$:

$$u = u_D, \tag{4.2.1}$$

2. Neumann BC on $\Gamma_N \subset \partial\Omega$:

$$\boldsymbol{n} \cdot (d\nabla u) = g_N, \tag{4.2.2}$$

3. Robin BC on $\Gamma_R \subset \partial\Omega$:

$$\boldsymbol{n} \cdot (d\nabla u) = g_R(u_\infty - u), \tag{4.2.3}$$

where $u_D : \Gamma_D \mapsto \mathbb{R},\ g_N : \Gamma_N \mapsto \mathbb{R},\ g_R : \Gamma_R \mapsto \mathbb{R}$ are prescribed functions, $\boldsymbol{n}$ is the outer normal on the surface $\partial\Omega$, and $u_\infty \in \mathbb{R}$ is a constant far-field solution, for example the temperature at infinity.

4.3 Variational Problems

Consider the following boundary value problem (BVP): find $u \in C^2(\Omega)$ such that

$$\begin{cases} L(u) &= f, & \text{in}\,\Omega \\ u &= u_D, & \text{on}\,\Gamma_D \\ \boldsymbol{n} \cdot (d\nabla u) &= g_N, & \text{on}\,\Gamma_N, \end{cases} \tag{4.3.1}$$

with $\partial\Omega = \Gamma_D \cup \Gamma_N$. In this general form, problem (4.3.1) is called the strong formulation of the BVP. Solutions of (4.3.1) are called strong or classical solutions.

Alternatively, FEMs attempt to solve a so-called weak formulation of the BVP, or just the variational problem. The variational problem corresponding to BVP (4.3.1) is to find $u \in H^1_{u_D}(\Omega; \Gamma_D)$ such that for all $v \in H^1_0(\Omega; \Gamma_D)$,

$$\int_\Omega \Big(d\nabla v \cdot \nabla u + v\big(\boldsymbol{c} \cdot \nabla u + s\, u - f\big)\Big) \mathrm{d}\Omega = \int_{\Gamma_N} v g_N \,\mathrm{d}\Gamma. \tag{4.3.2}$$

For definitions of the Sobolev spaces $H^1_{u_D}(\Omega; \Gamma_D)$ and $H^1_0(\Omega; \Gamma_D)$, the reader is referred to Appendix A.3.

Solutions to (4.3.2) are called weak solutions of (4.3.1). Weak solutions of (4.3.1) can be proven to exist and to be (local) unique [11, 24, 8]. Moreover, strong formulation (4.3.1) and variational problem (4.3.2) can be related by the following theorem [24]:

Theorem 4.3.1. *Provided that all functions are sufficiently smooth, problems (4.3.1) and (4.3.2) are equivalent.*

Proof: In this proof, the definition $C^p_g(\Omega;\Gamma) = \left\{u \in C^p(\Omega) \mid u\big|_\Gamma = g\right\}$ is adopted, for integer $p \in \mathbb{N}$, function $g : \Gamma \mapsto \mathbb{R}$ and subdomain $\Gamma \subset \partial\Omega$. Furthermore, it is assumed that $d \in C^1(\Omega) \subset L^\infty(\Omega)$ and $g_N \in L^2(\Gamma_N)$.

- (4.3.1) $\Longrightarrow$ (4.3.2):
  Suppose that $u \in C^2(\Omega)$ is a classical solution of (4.3.1). Let $v \in C^1_0(\Omega;\Gamma_D)$. Multiplying (4.1.1) by v and subsequently integrating over Ω yields

$$\int_\Omega v\Big(- \nabla\cdot(d\nabla u) + \boldsymbol{c}\cdot\nabla u + s\,u - f\Big)\,\mathrm{d}\Omega = 0. \tag{4.3.3}$$

  Partial differentiation of the first term in (4.3.3) gives

$$\int_\Omega \Big(- \nabla\cdot(dv\nabla u) + d\nabla v\cdot\nabla u + v(\boldsymbol{c}\cdot\nabla u + s\,u - f)\Big)\,\mathrm{d}\Omega = 0. \tag{4.3.4}$$

  Since both $d\nabla u$ and v are continuously differentiable, Gauss' divergence theorem can be applied to the first term in (4.3.4). The resulting integral over $\partial\Omega$ can be split up into

$$\int_{\partial\Omega} \boldsymbol{n}\cdot(dv\nabla u)\,\mathrm{d}\Gamma = \int_{\Gamma_D} \boldsymbol{n}\cdot(dv\nabla u)\,\mathrm{d}\Gamma + \int_{\Gamma_N} \boldsymbol{n}\cdot(dv\nabla u)\,\mathrm{d}\Gamma. \tag{4.3.5}$$

  The integral over Γ_D is zero, since $v\big|_{\Gamma_D} = 0$. So, application of Gauss'divergence theorem to (4.3.4) gives

$$\int_\Omega \Big(d\nabla v\cdot\nabla u + v(\boldsymbol{c}\cdot\nabla u + s\,u - f)\Big)\mathrm{d}\Omega = \int_{\Gamma_N} \boldsymbol{n}\cdot(dv\nabla u)\,\mathrm{d}\Gamma. \tag{4.3.6}$$

  Finally, substitution of the Neumann boundary condition (4.2.2) into the integral over Γ_N results in weak formulation (4.3.2). Thus, u is a solution of (4.3.2).

- (4.3.2) $\Longrightarrow$ (4.3.1):
  Suppose that $u \in C^2_{u_D}(\Omega;\Gamma_D)$ is a solution of (4.3.2). Assume that $\boldsymbol{c}$ and f are continuous in Ω. Then, application of partial differentiation to the first term of weak formulation (4.3.2) leads to

$$\int_\Omega \Big(\nabla\cdot\Big(dv\nabla u\Big) + v\Big(- \nabla\cdot(d\nabla u) + \boldsymbol{c}\cdot\nabla u + s\,u - f\Big)\Big)\mathrm{d}\Omega = \int_{\Gamma_N} v g_N\,\mathrm{d}\Gamma, \tag{4.3.7}$$

  for all $v \in H^1_0(\Omega;\Gamma_D)$. Since $d\nabla u$ is continuously differentiable, Gauss' divergence theorem can be applied to the first term of (4.3.7) for any $v \in C^1_0(\Omega;\Gamma_D)$. The result is

$$\int_\Omega v\Big(\underbrace{-\nabla\cdot(d\nabla u) + \boldsymbol{c}\cdot\nabla u + s\,u}_{=L(u)} - f\Big)\,\mathrm{d}\Omega \;=\; \int_{\Gamma_N} v\Big(g_N - \boldsymbol{n}\cdot(d\nabla u)\Big)\,\mathrm{d}\Gamma, \tag{4.3.8}$$

  for all $v \in C^1_0(\Omega;\Gamma_D)$. In order to prove that u solves (4.3.1), it is sufficient to prove that both the left-hand side integral and the right-hand side integral of (4.3.8) vanish for some $v \in C^1_0(\Omega;\Gamma_D)$. First, suppose that $(L(u) - f)(\boldsymbol{x}_0) \neq 0$ for some $\boldsymbol{x}_0 \in \Omega$. Without loss of generality, assume $(L(u) - f)(\boldsymbol{x}_0) > 0$. Note that because of the smoothness assumptions $L(u) - f$ is continuous in Ω. Then, let $\epsilon > 0$ be small enough, such that $L(u) - f > 0$ in the ball $\mathbb{B}(\boldsymbol{x}_0, \epsilon) \subset \Omega$. Choose $v \in C_0^1(\Omega; \Gamma_D)$, such that $v \geq 0$ in Ω and supp(v) $\subset \mathbb{B}(\boldsymbol{x}_0, \epsilon)$, with $v(\boldsymbol{x}_0) > 0$. Then, integration over Ω implies

$$\int_\Omega v(L(u) - f)\, \mathrm{d}\Omega > 0. \tag{4.3.9}$$

On the other hand, integral equation (4.3.8) leaves

$$\int_\Omega v(L(u) - f)\, \mathrm{d}\Omega = 0. \tag{4.3.10}$$

This is a contradiction, so $L(u) - f = 0$ in Ω. Next, suppose that $\big(g_N - \boldsymbol{n}\cdot(d\nabla u)\big)(\boldsymbol{x}_0) \neq 0$ for some $\boldsymbol{x}_0$ on Γ_N. Without loss of generality, assume $\big(g_N - \boldsymbol{n}\cdot(d\nabla u)\big)(\boldsymbol{x}_0) > 0$. Then, let $\epsilon > 0$ be small enough, such that $g_N - \boldsymbol{n}\cdot(d\nabla u) > 0$ on $\mathbb{B}(\boldsymbol{x}_0, \epsilon) \cap \Gamma_N$. Choose $v \in C_0^1(\Omega; \Gamma_D)$, such that $v \geq 0$ on Γ_N and supp(v) $\subset \mathbb{B}(\boldsymbol{x}_0, \epsilon)$, with $v(\boldsymbol{x}_0) > 0$. Then, integration over Γ_N implies

$$\int_{\Gamma_N} v\big(g_N - \boldsymbol{n}\cdot(d\nabla u)\big)\, \mathrm{d}\Gamma > 0. \tag{4.3.11}$$

This is in contradiction to (4.3.8). It follows that also (4.2.2) is satisfied. Thus, u is a classical solution of (4.3.1).

For a (variational) convection-diffusion problem, two kinds of BCs can be distinguished.

Essential boundary conditions Essential BCs prescribe the solution on some part of the boundary domain. The Dirichlet BC (4.2.1) is an essential BC; it is not incorporated in integral equation (4.3.2).

Natural boundary conditions Natural BCs do not need to be prescribed explicitly, since they are naturally included into the variational problem. A natural BC often concerns a surface flux or a boundary load. The Neumann BC (4.2.2) is a natural BC; from the second part of the proof of Theorem 4.3.1, it follows that the Neumann BC is satisfied by integral equation (4.3.2).

4.4 Linear Vector Spaces

Rather than discretising the differential operator (4.1.2), FEMs approximate the solution via a suitable choice of a finite dimensional subspace of $H^1(\Omega)$. Let $\{\phi_i\}_{i=1}^p \subset \mathcal{P}_q(\Omega)$ be a set of p linearly independent polynomials, then the span of $\{\phi_i\}_{i=1}^p$ is the p-dimensional, linear vector space

$$V^p(\Omega) := [\{\phi_i\}_{i=1}^p] = \Big\{v = \sum_{i=1}^p \phi_i v_i \;\Big|\; v_i \in \mathbb{R},\ i = 1, \ldots, p\Big\}. \tag{4.4.1}$$

Because the functions $\{\phi_i\}_{i=1}^p$ form a basis of $V^p(\Omega)$, they are called basis functions of $V^p(\Omega)$.

Similarly to (4.4.1), finite dimensional subspaces of the Sobolev space of weak solutions $H^1_{u_D}(\Omega;\Gamma_D)$ and the Sobolev space of weight functions $H^1_0(\Omega;\Gamma_D)$ can be introduced. Let $p \in \mathbb{N}$. Then, a linearly independent set of suitable basis functions $\{\phi_i\}_{i=1}^p \subset H^1(\Omega)$ gives rise to the linear vector spaces

$$U_{u_D}(\Omega;\Gamma_D) := \left\{\hat{u} = \sum_{i=1}^{p} \phi_i \hat{u}_i \;\middle|\; \hat{u}\Big|_{\Gamma_D} = u_D,\ \hat{u}_i \in \mathbb{R},\ i = 1,\dots,p\right\} \subset H^1_{u_D}(\Omega;\Gamma_D), \tag{4.4.2}$$

$$W_0(\Omega;\Gamma_D) := \left\{\hat{v} = \sum_{i=1}^{p} \phi_i \hat{v}_i \;\middle|\; \hat{v}\Big|_{\Gamma_D} = 0,\ \hat{v}_i \in \mathbb{R},\ i = 1,\dots,p\right\} \subset H^1_0(\Omega;\Gamma_D), \tag{4.4.3}$$

Note that in order to construct the subspace $U_{u_D}(\Omega;\Gamma_D)$, it is required that $u_D \in [\{\phi_i\}_{i=1}^p]$ in an approximate sense.

4.5 Galerkin Finite Element Discretisation

Consider the weak formulation of BVP (4.3.1): find $u \in H^1_{u_D}(\Omega;\Gamma_D)$ such that

$$\forall_{v \in H^1_0(\Omega;\Gamma_D)} \;:\; a(v,u) = l(v), \tag{4.5.1}$$

where $a : H^1_0(\Omega;\Gamma_D) \times H^1_{u_D}(\Omega;\Gamma_D) \mapsto \mathbb{R}$ and $l : H^1_0(\Omega;\Gamma_D) \mapsto \mathbb{R}$ are forms, defined by

$$a(v,u) = \int_\Omega \Big(d\nabla v \cdot \nabla u + v(\boldsymbol{c} \cdot \nabla u + s\, u)\Big)\mathrm{d}\Omega, \tag{4.5.2}$$

$$l(v) = \int_\Omega v f \;\mathrm{d}\Omega + \int_{\Gamma_N} v g_N \;\mathrm{d}\Gamma. \tag{4.5.3}$$

Note that form (4.5.2) is bilinear in (v,u) and form (4.5.3) is linear in v. Moreover, if $\boldsymbol{c} = \boldsymbol{0}$, then (4.5.2) is symmetric, that is, $a(v,u) = a(u,v)$.

The Galerkin approximation of variational problem (4.5.1) is to find $\hat{u} \in U_{u_D}(\Omega;\Gamma_D)$ such that

$$\forall_{\hat{v} \in W_0(\Omega;\Gamma_D)} \;:\; a(\hat{v},\hat{u}) = l(\hat{v}). \tag{4.5.4}$$

Let $\{\phi_i\}_{i=1}^p$ be the set of basis functions that spans $U_{u_D}(\Omega;\Gamma_D)$ and $W_0(\Omega;\Gamma_D)$. Then, any $\hat{u} \in U_{u_D}(\Omega;\Gamma_D)$ and $\hat{v} \in W_0(\Omega;\Gamma_D)$ can be written as

$$\hat{u} = \boldsymbol{\phi}^T \underset{\sim}{\hat{\boldsymbol{u}}}, \qquad \hat{v} = \boldsymbol{\phi}^T \underset{\sim}{\hat{\boldsymbol{v}}}, \tag{4.5.5}$$

with

$$\underset{\sim}{\hat{\boldsymbol{u}}} = \begin{pmatrix} \hat{u}_1 \\ \vdots \\ \hat{u}_p \end{pmatrix}, \qquad \underset{\sim}{\hat{\boldsymbol{v}}} = \begin{pmatrix} \hat{v}_1 \\ \vdots \\ \hat{v}_p \end{pmatrix}, \qquad \boldsymbol{\phi} = \begin{pmatrix} \phi_1 \\ \vdots \\ \phi_p \end{pmatrix}. \tag{4.5.6}$$

Discretised problem (4.5.4) restricts variational problem (4.5.1) to finding the vector of function values $\underset{\sim}{\hat{\boldsymbol{u}}}$. Therefore, it is convenient to write (4.5.4) in terms of (4.5.6). To this end, the derivatives of $\hat{u}$ and $\hat{v}$ are written as linear combinations of the derivatives of the basis functions:

$$\nabla\hat{u} = (\nabla\boldsymbol{\phi})^T\underset{\sim}{\hat{\boldsymbol{u}}}, \qquad \nabla\hat{v} = (\nabla\boldsymbol{\phi})^T\underset{\sim}{\hat{\boldsymbol{v}}}. \tag{4.5.7}$$

Substitution of (4.5.5) and (4.5.7) into (4.5.4) gives

$$a(\hat{v},\hat{u}) = \underset{\sim}{\hat{\boldsymbol{v}}}^T\boldsymbol{\mathcal{K}}\underset{\sim}{\hat{\boldsymbol{u}}}, \tag{4.5.8}$$

$$l(\hat{v}) = \underset{\sim}{\hat{\boldsymbol{v}}}^T\boldsymbol{q}, \tag{4.5.9}$$

where

$$\boldsymbol{\mathcal{K}} = \int_\Omega \Big(d\nabla\boldsymbol{\phi}(\nabla\boldsymbol{\phi})^T + \boldsymbol{\phi}\cdot(\boldsymbol{c}\cdot\nabla\boldsymbol{\phi} + s\,\boldsymbol{\phi})\Big)\mathrm{d}\Omega, \tag{4.5.10}$$

$$\boldsymbol{q} = \int_\Omega \boldsymbol{\phi} f\,\mathrm{d}\Omega + \int_{\Gamma_N}\boldsymbol{\phi} g_N\,\mathrm{d}\Gamma, \tag{4.5.11}$$

Equation (4.5.4) should hold for any vector $\underset{\sim}{\hat{\boldsymbol{v}}} \in \mathbb{R}^p$, such that $\boldsymbol{\phi}^T\underset{\sim}{\hat{\boldsymbol{v}}} = 0$ on Γ_D, hence

$$\boldsymbol{\mathcal{K}}\underset{\sim}{\hat{\boldsymbol{u}}} = \boldsymbol{q}. \tag{4.5.12}$$

System of equations (4.5.12) is the Galerkin approximation of variational problem (4.5.1). In a Galerkin approximation, the vector spaces $U_{u_D}(\Omega;\Gamma_D)$ and $W_0(\Omega;\Gamma_D)$ are spanned by the same set of basis functions. The matrix $\boldsymbol{\mathcal{K}}$ is often referred to as the stiffness matrix and $\boldsymbol{q}$ is the load vector. Note that

$$\mathcal{K}_{ij} = a(\phi_i,\phi_j), \qquad q_i = l(\phi_i), \qquad \text{for}\, i,j = 1,\ldots,p. \tag{4.5.13}$$

In general, it is required that the number of basis functions p is quite large, in order to obtain a sufficient accurate approximation of the solution of (4.5.1). If the matrix $\boldsymbol{\mathcal{K}}$ is full, existing methods to solve system of equations (4.5.12) can become rather inefficient. Therefore, it is of major importance to construct a set of basis functions, such that solving (4.5.12) is considerably efficient. The next subsection discusses some important properties that the basis functions should satisfy, in order to reduce the amount of such calculations.

4.6 Characteristic Properties of the Basis Functions

In the previous subsection, variational problem (4.3.2) is approximated by means of a finite set of linearly independent basis functions $\{\phi_i\}_{i=1}^p$, which span p-dimensional linear vector spaces for approximations of the weak solutions. The question remains how to choose a set of basis functions, such that system of equations (4.5.12) can be solved in an efficient way.

In order to reduce the amount of calculations required to solve system of equations (4.5.12), it is desired that the stiffness matrix (4.5.10) is sparse. Typically, sparse matrices have regular patterns of block diagonals with nonzero indices, whereas the other indices are zero. So, a set of basis functions is looked for that satisfies $a(\phi_i, \phi_j) = 0$ apart from a regularly structured sparsity pattern.

The construction of an appropriate set of basis functions can be illustrated by means of an example for the one dimensional case [19].

Example 4.6.1. *Let $\Omega = (0, 1)$, $N \in \mathbb{N}$. Subdivide Ω into line segments $e_i = (\hat{x}_i, \hat{x}_{i+1})$, for $i = 1, \dots N$, with nodal points $\hat{x}_j = \frac{j-1}{N}$, for $j = 1, \dots N + 1$. Then, construct a set of basis functions $\{\phi\}_{i=1}^{N+1}$, where ϕ_i is defined by*

$$\phi_1(x) = \begin{cases} 1 - Nx, & \text{on } e_1; \\ 0, & \text{elsewhere}, \end{cases}$$

$$\phi_j(x) = \begin{cases} 1 + N(x - \hat{x}_j), & \text{on } e_{j-1}; \\ 1 - N(x - \hat{x}_j), & \text{on } e_j; \\ 0, & \text{elsewhere}, \end{cases} \qquad \text{for } j = 2, \dots, N,$$

$$\phi_{N+1}(x) = \begin{cases} 1 + N(x - 1), & \text{on } e_N; \\ 0, & \text{elsewhere}, \end{cases}$$

Note that the basis functions satisfy

$$\phi_i(\hat{x}_j) = \begin{cases} 1, & \text{if } i = j; \\ 0, & \text{else}. \end{cases}$$

Therefore, the nodal point $\hat{x}_i$ is called a support point of ϕ_i, for $i = 1, \dots, N + 1$. Moreover,

$$\sum_i \phi_i(x) = 1,$$

for all $x \in \overline{\Omega}$ and the basis functions are linear per element. In Figure 4.1 a plot of $\phi_3(x)$ and $\phi_4(x)$ for $N = 7$ is given. From this figure, it can be seen that for the stiffness matrix (4.5.10), a typical sparsity pattern as in Figure 4.2 is obtained.

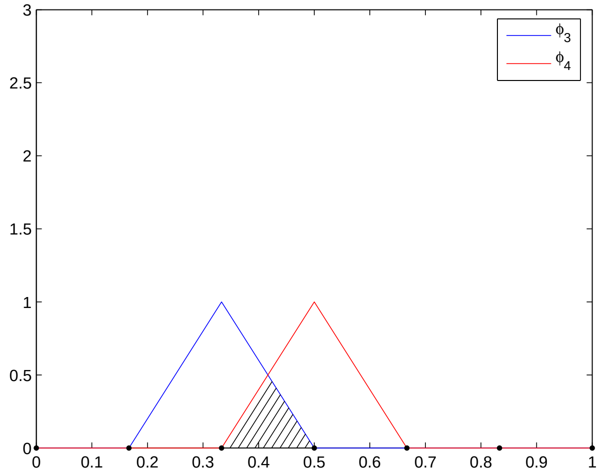


Figure 4.1: Illustration of one dimensional basis functions and the support of their product.

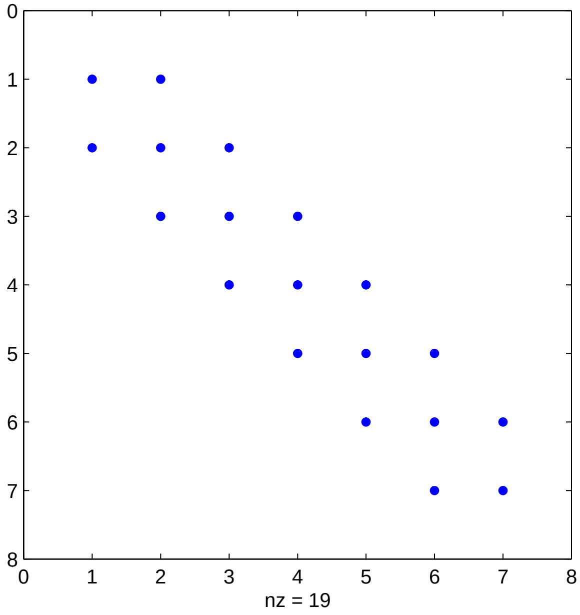


Figure 4.2: Sparsity pattern for the one dimensional basis functions. The corresponding 8 × 8 stiffness matrix contains only 19 non-zero entries.

Example 4.6.1 can be generalised in order to construct a basis for problem (4.5.4). Let $N \in \mathbb{N}$. Partition the domain of interest Ω into so-called elements e_i, $i = 1, \ldots, N$, i. e.

$$\overline{\Omega} = \overline{\bigcup_{i=1}^{N}} e_i. \tag{4.6.1}$$

Assume that the elements are open subdomains of Ω and denote by ∂e_i the boundary of element e_i, $i = 1, \ldots, N$. For some $p \in \mathbb{N}$, let $\{\hat{\boldsymbol{x}}_j\}_{j=1}^{p} \subset \overline{\Omega}$ be the set of nodal points (e.g. vertices) of the elements in Ω. A nodal point $\hat{\boldsymbol{x}}_j$ is said to be a boundary node of an element e_i, if

$$\hat{\boldsymbol{x}}_j \in \partial e_i, \qquad i = 1, \ldots, N, \; j = 1, \ldots, p.$$

On the other hand, $\hat{\boldsymbol{x}}_j$ is said to be a internal node of e_i, if

$$\hat{\boldsymbol{x}}_j \in e_i, \qquad i = 1, \ldots, N, \; j = 1, \ldots, p.$$

The element boundaries ∂e_i, $i = 1, \ldots, N$, can be subdivided into edges between the boundary nodes. An edge from boundary node $\hat{\boldsymbol{x}}_k$ to boundary node $\hat{\boldsymbol{x}}_l$ is denoted by $\partial(\hat{\boldsymbol{x}}_k, \hat{\boldsymbol{x}}_l)$. For instance, Figure 4.3 illustrates a quadrilateral element with 5 nodes and 4 edges. Subsequently, define a set of basis functions $\{\phi_j\}_{j=1}^{p}$, such that for all $i = 1, \ldots, p$, the following properties are satisfied:

$$\phi_j(\hat{\boldsymbol{x}}_j) = \delta_{jk} = \begin{cases} 1, & \text{if } i = j; \\ 0, & \text{else,} \end{cases} \tag{4.6.2}$$

$$\overline{\text{supp}(\phi_\text{j})} = \overline{\bigcup \left\{ e_i \mid \hat{\boldsymbol{x}}_j \in \overline{e}_i \right\}}, \tag{4.6.3}$$

$$\phi_j = 0, \qquad \text{on } \partial(\hat{\boldsymbol{x}}_k, \hat{\boldsymbol{x}}_l), \quad j \neq k, l. \tag{4.6.4}$$

Property (4.6.2) defines the support points corresponding to the basis functions, that is if property (4.6.2) is satisfied, then $\hat{\boldsymbol{x}}_j$ is called a support point of ϕ_j, for $j = 1, \ldots, p$. Property (4.6.3) is illustrated in Figure 4.4, in which a 2D square domain Ω is uniformly partitioned into square elements. Here, the grey area is the support of ϕ_{25}. The nodes and elements are numbered from left to right, bottom to top. In this figure, node 25 is the support point of ϕ_{25}. Finally, condition (4.6.4) says that the basis function is zero on the boundary of the support. Properties (4.6.2)-(4.6.4) are characteristic for FEMs. They induce sparsity of the stiffness matrix, as $a(\phi_i, \phi_j)$ is only nonzero if $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ belong to the same element.

Obviously, a construction of a set of basis functions satisfying properties (4.6.2) and (4.6.3) attends a partition into elements. As a consequence, it may not be easy to construct a suitable set of basis functions, if the elements have complex geometries. Therefore, it can be useful to introduce local transformations of the geometry in order to construct basis functions at element level. To this purpose, so-called isoparametric elements are introduced. This is the topic of Section 4.8.

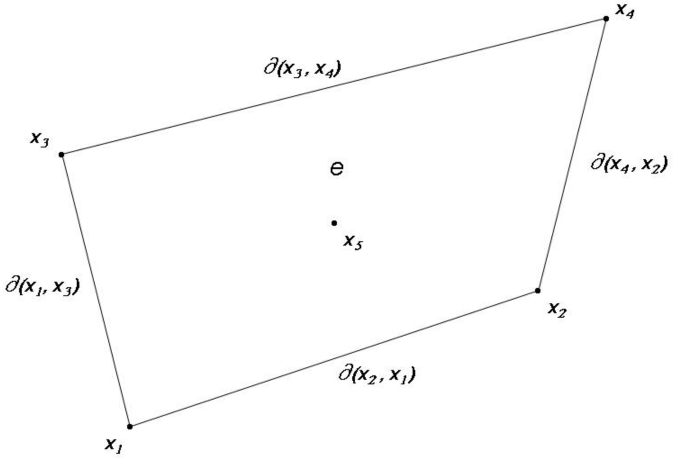


Figure 4.3: A quadrilateral element with 5 nodes and 4 edges. The four vertices are boundary nodes and the node in $\boldsymbol{x}_5$ is an internal node.

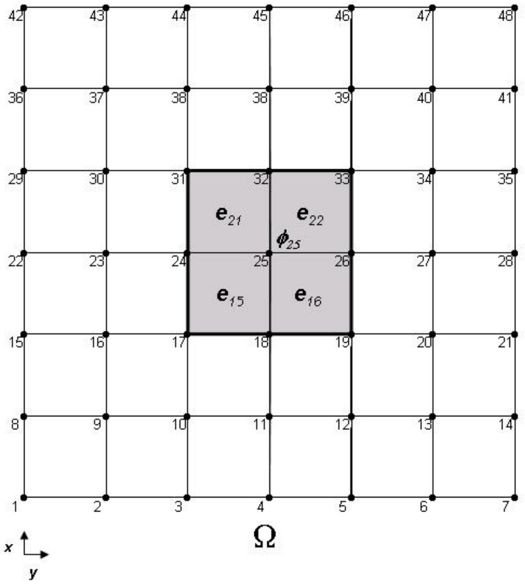


Figure 4.4: Support of a basis function for a square partition of a squared domain Ω into square elements.

4.7 Assembly of the Stiffness Matrix and the Load Vector

The stiffness matrix $\mathcal{K}$ and the load vector $\boldsymbol{q}$ can be obtained from the contribution of each element. For each element e, let l_e be the number of nodes of e and the vector $\boldsymbol{\phi}_e$ denote the vector comprising the l_e corresponding basis functions. Then, $\mathcal{K}$ and $\boldsymbol{q}$ can be assembled by a summation of the contributions of different elements:

$$\mathcal{K} = \sum_e \mathcal{P}_e \mathcal{K}_e \mathcal{P}_e^T, \tag{4.7.1}$$

$$\boldsymbol{q} = \sum_e \mathcal{P}_e \boldsymbol{q}_e, \tag{4.7.2}$$

where, for all elements e,

$$\mathcal{K}_e = \int_e \Big(d \nabla \boldsymbol{\phi}_e (\nabla \boldsymbol{\phi}_e)^T + \boldsymbol{\phi}_e \cdot (\boldsymbol{c} \cdot \nabla \boldsymbol{\phi}_e + s\, \boldsymbol{\phi}_e) \Big) \mathrm{d}\Omega, \tag{4.7.3}$$

$$\boldsymbol{q}_e = \int_e \boldsymbol{\phi}_e f \,\mathrm{d}\Omega + \int_{e \cap \Gamma_N} \boldsymbol{\phi}_e g_N \,\mathrm{d}\Gamma. \tag{4.7.4}$$

and $\mathcal{P}_e$ is a $p \times l_e$ projection matrix that maps the entries of the $l_e \times l_e$ element stiffness matrix $\mathcal{K}_e$ and $l_e \times 1$ element load vector $\boldsymbol{q}_e$ onto the entries of the global matrix $\mathcal{K}$ and load vector $\boldsymbol{q}$, respectively. So for any element e, the indices of $\mathcal{P}_e$ are given by

$$P_{e,ij} = \begin{cases} 1, & \text{if } \boldsymbol{x}_i = \boldsymbol{x}_{e,j}; \\ 0, & \text{else}, \end{cases} \tag{4.7.5}$$

where $\boldsymbol{x}_{e,j}$ is the j^{th} node of element e, for $j = 1, \ldots, l_e$. Expressions (4.7.1) and (4.7.2) are often used in computer implementations of FEMs.

4.8 Isoparametric Elements

In Section 4.6, a construction of a set of basis functions by means of a partition into elements is proposed. In such a construction, the definitions of the set of basis functions and the mesh are closely connected to each other. In the present section, it is shown how the basis functions can be constructed by a suitable mapping of the element geometry.

Before going into more detail, it is worth studying some terminology.

Mesh: a partition into elements (see Figure 4.5).

Element type: the set of nodes, numbers of unknowns per node and edges of an element. Two elements are of the same type, if there is a bijective mapping of the set of nodes, edges and unknowns of the elements onto each other.

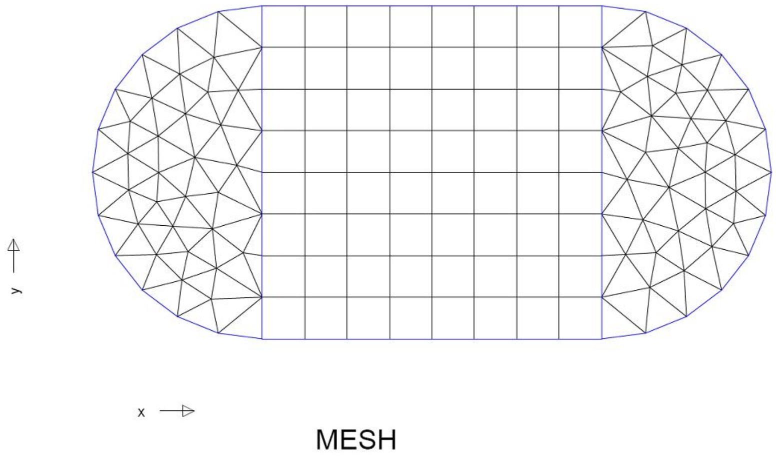


Figure 4.5: A mesh consisting of three submeshes.

Submesh: a part of a mesh that is a partition into elements of the same type. Each mesh can be subdivided into different submeshes (see Figure 4.5).

Reference element: an element that represents the elements in a submesh and has a relatively simple geometry. Usually, computations are carried out for the reference element and solutions are mapped onto the elements in the submesh.

Since the basis functions are constructed at element level, it is convenient to introduce a local mapping of the element geometry, such that a straightforward, local construction of the basis functions is possible. A local mapping of the geometry at element level can be achieved by mapping a reference element onto an element in the submesh. This mapping is the result of interpolation over the nodes of the reference element. Consider an element e, with nodes $\hat{\boldsymbol{x}}_1, \ldots, \hat{\boldsymbol{x}}_{l_e}$. Let $\hat{e}$ be a reference element of the same type, with nodes $\hat{\boldsymbol{\xi}}_1, \ldots, \hat{\boldsymbol{\xi}}_{l_e}$, and let $\{\varphi\}_{i=1}^{l_e} \subset H^1(\hat{e})$ be a set of interpolation functions corresponding to the nodes $\hat{\boldsymbol{\xi}}_1, \ldots, \hat{\boldsymbol{\xi}}_{l_e}$ of $\hat{e}$, such that

$$\begin{aligned} \varphi_i(\hat{\boldsymbol{\xi}}_j) &= \delta_{ij}, \qquad \text{for}\, i, j = 1, \ldots, l_e, \\ \varphi_j(\boldsymbol{\xi}) &= 0, \qquad \text{for}\, \boldsymbol{\xi} \in \partial(\hat{\boldsymbol{x}}_k, \hat{\boldsymbol{x}}_l), \quad \text{if}\ j \neq k, l. \end{aligned}$$

Write

$$\boldsymbol{\varphi} = \begin{pmatrix} \varphi_1 \\ \vdots \\ \varphi_{l_e} \end{pmatrix}.$$

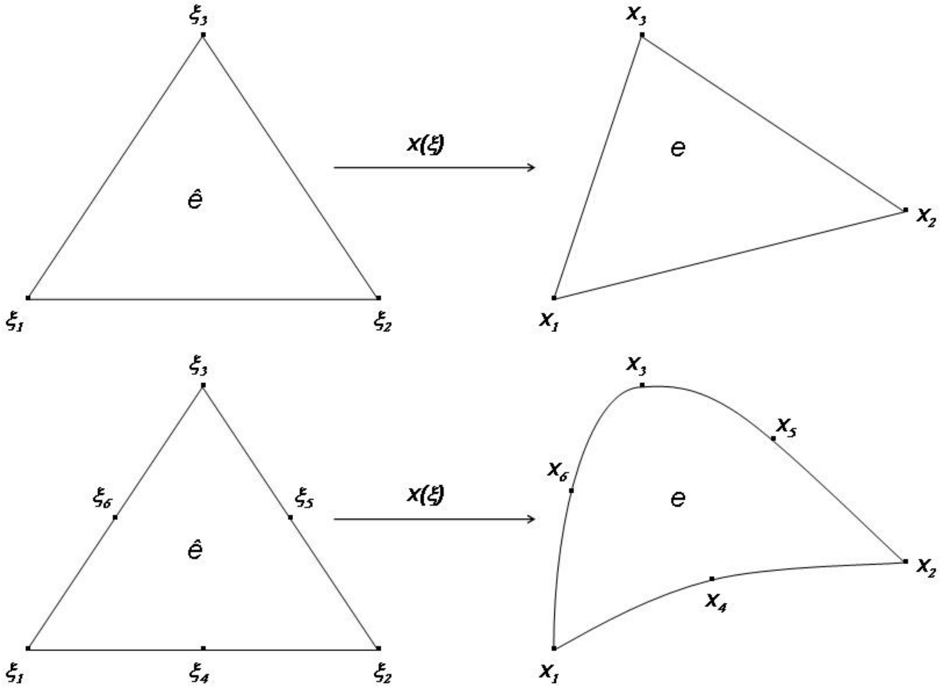


Figure 4.6: Mapping of reference elements onto arbitrary elements of equal topology.

Then, the mapping of the geometry of $\hat{e}$ onto e is given by the coordinate interpolation

$$\boldsymbol{x}(\boldsymbol{\xi}) = \boldsymbol{X}_e\boldsymbol{\varphi}(\boldsymbol{\xi}), \qquad \text{for}\, \boldsymbol{\xi} \in \hat{e}, \tag{4.8.1}$$

where $\boldsymbol{X}_e = (\hat{\boldsymbol{x}}_1|\cdots|\hat{\boldsymbol{x}}_{l_e})$. Below, some examples for (bi-)linear interpolation functions are given.

Example 4.8.1. *Consider a domain partition into two-dimensional quadrilateral elements e_i, $i = 1,\dots,p$. In this case, a suitable reference element would be the square $\hat{e} = (-1,1)^2$. Denote the nodes of $\hat{e}$ by $(\hat{\xi}_j,\hat{\eta}_j)$, for $j = 1,\dots,4$. Then, for each $i = 1,\dots,p$, a local mapping of $\hat{e}$ onto e_i is given by*

$$\boldsymbol{x}(\xi,\eta) = \boldsymbol{X}_i\boldsymbol{\varphi}(\xi,\eta), \qquad \mathit{for}\, (\xi,\eta) \in \hat{e},$$

where $\boldsymbol{X}_i$ comprises the nodes of element e_i and $\boldsymbol{\varphi} : \hat{e} \mapsto \mathbb{R}$ is a vector function, with φ_j defined by

$$\varphi_j(\xi,\eta) = \frac{1}{4}(1+\hat{\xi}_j\xi)(1+\hat{\eta}_j\eta), \qquad j = 1,2,3,4. \tag{4.8.2}$$

Example 4.8.2. *Consider two-dimensional triangular elements e_i, $i = 1, \dots, p$. Let $\hat{e}$ be a triangular reference element with nodes $(\hat{\xi}_j, \hat{\eta}_j)$, for $j = 1, 2, 3$. This leads to a local mapping of $\hat{e}$ onto e_i, for each $i = 1, \dots, p$, given by*

$$\boldsymbol{x}(\xi, \eta) = \boldsymbol{X}_i \boldsymbol{\varphi}(\xi, \eta), \qquad \textit{for}\, \boldsymbol{\xi} \in \hat{e},$$

where $\boldsymbol{X}_i$ comprises the nodes of element e_i and $\boldsymbol{\varphi} : \hat{e} \mapsto \mathbb{R}$ has entries φ_j defined by

$$\varphi_j(\xi, \eta) = \frac{1}{\lambda} \sum_{k \neq j} \sum_{l \neq j,k} e_{jkl}(\hat{\xi}_k - \xi)(\hat{\eta}_l - \eta), \qquad j = 1, \dots, 3. \tag{4.8.3}$$

where

$$\lambda = \sum_j \sum_{k \neq j} \sum_{l \neq j,k} e_{jkl} \hat{\xi}_k \hat{\eta}_l, \tag{4.8.4}$$

with e_{jkl} the alternator, which is given by

$$e_{jkl} = \begin{cases} 0, & \textit{if two indices are equal}; \\ 1, & \textit{if } j, k, l \textit{ is a cyclic permutation}; \\ -1, & \textit{if } j, k, l \textit{ is a non-cyclic permutation}. \end{cases} \tag{4.8.5}$$

Suitable reference elements are for example equilateral or right-angled triangles.

An element e is said to be isoparametric, if both spatial coordinates and finite element solutions are approximated by means of the same set of basis functions (F. P. T. Baaijens, [3]). So for isoparametric elements, the vector of basis functions $\boldsymbol{\phi}_e$ in element e is defined as

$$\boldsymbol{\phi}_e(\boldsymbol{x}) = \begin{cases} \boldsymbol{\varphi}(\boldsymbol{\xi}), & \text{with } \boldsymbol{x} \equiv \boldsymbol{x}(\boldsymbol{\xi}), \text{ for } \boldsymbol{\xi} \in \hat{e}, \\ 0, & \text{else}. \end{cases} \tag{4.8.6}$$

Then, $\hat{u}$ can be written in terms of $\boldsymbol{\xi}$ by means of (4.8.6),

$$\hat{u}(\boldsymbol{x}) = (\boldsymbol{\phi}_e(\boldsymbol{x}))^T \underset{\sim}{\hat{\boldsymbol{u}}}_e = \left(\boldsymbol{\varphi}(\boldsymbol{\xi})\right)^T \underset{\sim}{\hat{\boldsymbol{u}}}_e, \qquad \text{with } \boldsymbol{x} \equiv \boldsymbol{x}(\boldsymbol{\xi}), \text{ for } \boldsymbol{\xi} \in \hat{e}, \tag{4.8.7}$$

where $\underset{\sim}{\hat{\boldsymbol{u}}}_e$ comprises the solution values corresponding to the vector of basis functions $\boldsymbol{\phi}_e$. Clearly, $\boldsymbol{\varphi}$ is the vector of basis functions in $\hat{e}$. For isoparametric elements, the finite element solutions can directly be computed by means of the basis functions in $\hat{e}$. Moreover, the finite element solutions are approximated in the same way as the geometry.

For isoparametric elements, the element stiffness matrix and element load vector can be evaluated in terms of the nodes of $\hat{e}$. Suppose that a local transformation of the geometry is applied on an element $e \subset \Omega$. Then, the element matrices (4.7.3) and (4.7.4) can be written in terms of $\boldsymbol{\varphi}$. To this purpose, apply the chain rule for differentiation to obtain a relation between the Jacobian matrices of $\boldsymbol{\phi}$ and $\boldsymbol{\varphi}$:

$$\nabla_{\boldsymbol{x}} \boldsymbol{\phi}_e = \nabla_{\boldsymbol{x}} \boldsymbol{\xi} \; \nabla_{\xi} \boldsymbol{\varphi}, \tag{4.8.8}$$

where the subscripts $_{\boldsymbol{x}}$ and $_{\xi}$ indicate to which variables differentiation or integration is applied. Here, $\nabla_{\boldsymbol{x}} \boldsymbol{\xi}$ is found from the inverse of $\nabla_{\xi} \boldsymbol{x}$:

$$\nabla_{\boldsymbol{x}} \boldsymbol{\xi} = (\nabla_{\xi} \boldsymbol{x})^{-1} = (\nabla_{\xi} \boldsymbol{\varphi} \, \boldsymbol{X}_e^T)^{-1}. \tag{4.8.9}$$

Finally, the element stiffness matrix and element load vector can be calculated as

$$\mathcal{K}_e = \int_{\hat{e}} \Big(d (\nabla_{\xi} \boldsymbol{x})^{-1} \, \nabla_{\xi} \boldsymbol{\varphi} \left((\nabla_{\xi} \boldsymbol{x})^{-1} \, \nabla_{\xi} \boldsymbol{\varphi} \right)^T + \boldsymbol{\varphi} \cdot \left(\boldsymbol{c} \cdot (\nabla_{\xi} \boldsymbol{x})^{-1} \, \nabla_{\xi} \boldsymbol{\varphi} + s \, \boldsymbol{\varphi} \right) \Big) \left| \nabla_{\xi} \boldsymbol{x} \right| \, \mathrm{d}\Omega_{\xi}, \tag{4.8.10}$$

$$\boldsymbol{q}_e = \int_{\hat{e}} \boldsymbol{\varphi} f \left| \nabla_{\xi} \boldsymbol{x} \right| \, \mathrm{d}\Omega_{\xi} + \int_{\partial \hat{e}_N} \boldsymbol{\varphi} g_N \left| \nabla_{\xi} \boldsymbol{x} \Big|_{\partial \hat{e}_N} \right| \, \mathrm{d}\Gamma_{\xi}, \tag{4.8.11}$$

where $\partial \hat{e}_N$ denotes the mapping of $e \cap \Gamma_N$ onto $\hat{e}$, and $\nabla_{\xi} \boldsymbol{x} \Big|_{\partial \hat{e}_N}$ is the Jacobian matrix of $\boldsymbol{x}(\boldsymbol{\xi})$ restricted to $\partial \hat{e}_N$.

4.9 Numerical Integration

In general, integrals (4.8.10) and (4.8.11) cannot be integrated analytically. However, they can be approximated by using a numerical integration formula. A suitable numerical integration scheme can be obtained by applying an interpolation formula. Suppose a function $F \equiv F(\boldsymbol{\xi})$ is defined on an element $\hat{e}$. Let $\{\boldsymbol{\xi}_i\}_{i=1}^{q} \subset \overline{\hat{e}}$ be a set of interpolation points, where $q \in \mathbb{N}$ is the so-called interpolation order. Then, apply interpolation on F,

$$F \approx \sum_{i=1}^{q} p_i F_i, \tag{4.9.1}$$

for some interpolation functions $p_1, \ldots, p_q$. Here, $F_i = F(\boldsymbol{\xi}_i)$, for $i = 1, \ldots, q$. Then, integration yields a quadrature formula,

$$\int_{\hat{e}} F \mathrm{d}\Omega_{\xi} \approx \sum_{i=1}^{q} w_i F_i, \tag{4.9.2}$$

where w_i are the weights, which are given by

$$w_i = \int_{\hat{e}} p_i \mathrm{d}\Omega_{\xi}, \qquad \text{for } i = 1, \ldots, q. \tag{4.9.3}$$

In this context, the interpolation points are also referred to as quadrature points.

The choice of the quadrature points influences the accuracy of the quadrature formula. Firstly, the higher the interpolation order, the higher the order of the polynomial approximation of the integral. In fact, it can be proven that the quadrature formula is exact for polynomials of degree $\leq q-1$ [8]. Secondly, the accuracy of the quadrature formula is determined by the position of the quadrature points. Therefore, they should be chosen such as to obtain optimal accuracy. To this end, tables with numbers and positions of quadrature points for optimal accuracy are available for different element shapes [3, 25].

4.10 Numerical Boundary Conditions

In theory, essential BCs of variational problem (4.5.4) are enforced in the solution space, so that only solutions of linear system (4.5.12) are allowed that satisfy the essential BCs. However, in practice, there is no guarantee that numerical solutions of (4.5.12) satisfy the essential BCs. Therefore, the linear system should be provided with an additional constraint

$$\boldsymbol{\phi}^T \underset{\sim}{\hat{\boldsymbol{u}}} = u_D, \qquad \text{on}\, \Gamma_D. \tag{4.10.1}$$

So, for any $i = 1, \ldots, p$, such that $\boldsymbol{x}_i \in \Gamma_D$, the i^{th} vector index of $\underset{\sim}{\hat{\boldsymbol{u}}}$ should satisfy

$$\hat{u}_i = (\boldsymbol{\phi}(\hat{\boldsymbol{x}}_i))^T \underset{\sim}{\hat{\boldsymbol{u}}} = u_D(\hat{\boldsymbol{x}}_i). \tag{4.10.2}$$

Condition (4.10.2) can be included in linear system (4.5.12) as follows (J. M. L. Maubach, [19]). Define the projection matrix

$$\mathcal{P}_p := \operatorname{diag}(p_i), \tag{4.10.3}$$

where

$$p_i = \begin{cases} 1, & \text{if}\, \hat{\boldsymbol{x}}_i \in \Omega \cup \Gamma_N; \\ 0, & \text{if}\, \hat{\boldsymbol{x}}_i \in \Gamma_D, \end{cases} \qquad \text{for}\, i = 1, \ldots, p. \tag{4.10.4}$$

The projection matrix is used to modify the stiffness matrix and load vector. Define the matrices

$$\mathcal{K}_D = \mathcal{P}_p \mathcal{K} + (\mathcal{I}_p - \mathcal{P}_p), \tag{4.10.5}$$

$$\boldsymbol{q}_D = \mathcal{P}_p \boldsymbol{q} + (\mathcal{I}_p - \mathcal{P}_p)\, \underset{\sim}{\boldsymbol{u}}_D\,, \tag{4.10.6}$$

where $\underset{\sim}{\boldsymbol{u}}_D$ is the column comprising $u_D(\hat{\boldsymbol{x}}_1), \ldots, u_D(\hat{\boldsymbol{x}}_n)$. Then, the solution of linear system

$$\mathcal{K}_D\, \underset{\sim}{\hat{\boldsymbol{u}}} = \boldsymbol{q}_D \tag{4.10.7}$$

satisfies the Dirichlet BC (4.2.1).

Chapter 5

Finite Element Analysis of Stokes Flow Problems in Glass Pressing

In Chapter 4, FEMs for convection diffusion equations are studied. The level set equation and the heat equation in glass forming models are examples of convection diffusion equations. On the other hand, the Stokes flow equations are not covered by the family of convection diffusion equations.

Since the solver problem discussed in Chapter 1 is observed for incompressible Stokes flow problems only, this chapter focusses on the finite element theory for this particular class of problems. The purpose of this chapter is to show that an appropriate finite element discretisation of Stokes flow problems leads to a well-defined linear system of equations. That is, the discretisation method should enable one to solve the resulting system by means of an appropriate solver within a reasonable amount of time. Apart from a discussion which discretisation methods can be found most suitable for the Stokes flow problem, it is elucidated which methods are used in TNO Glass Group's pressing model, as it is sought to analyse the solver problem for this simulation model in particular. This chapter will make it clear that the finite element discretisation methods used in the pressing model lead to a theoretically well-defined system of equations. Then, the next step is to find a suitable numerical method to solve the discretised Stokes flow problem. This is the topic of the Chapter 6.

This chapter is structured as follows. First, section 5.1 gives a weak formulation for Stokes flow problems in pressing models. Then, section 5.2 discretises this weak formulation by means of Galerkin FEMs. Furthermore, section 5.3 shows under which conditions the weak formulation and the discretised problem have a unique solution, and if these conditions are satisfied for Stokes flow problems in a glass pressing model. Here, subsection 5.3.1 focusses on the weak formulation and subsection 5.3.2 on the discretised problem. In the discretised Stokes flow problem, a zero block in the coefficient matrix appears on the diagonal, as a consequence of the absence of a pressure term in the continuity equation. As this zero block can lead to considerable problems, section 5.4 discusses several solution methods to deal with this. In this section, the solution method used in TNO Glass

Group's pressing model is compared to others, and it is motivated why the same method is also applied in this thesis. Next, section 5.5 discusses and compares different stable elements for Stokes flow problems. The elements that are used in TNO Glass Group's pressing model are called Mini-elements. A Mini-element stabilises a Stokes flow problem by means of a so-called bubble-function. Section 5.6 shows for triangular Mini-elements that elimination of the bubble functions at element level leads to a stabilised formulation of Stokes flow problems, for which it can be proven that it has a unique solution. Since the finite element mesh used is crucial for the success of the FEM applied, section 5.7 explains how such a mesh is constructed in TNO Glass Group's pressing model. Finally, section 5.8 states the conclusions on this chapter.

5.1 Variational Stokes Flow Problem

Consider the dimensionless, incompressible Stokes flow problem in the axi-symmetrical pressing model in section 3.2: find $\boldsymbol{u} \in \left(C^2(\Omega)\right)^2$ and $p \in C^1(\Omega)$, such that

$$\begin{cases} \nabla \cdot (2\mu \nabla_s \boldsymbol{u})_d - \nabla p \;=\; \boldsymbol{0}, \quad \nabla \cdot \boldsymbol{u} \;=\; 0, & \text{in}\,\Omega, \\ \boldsymbol{u}_* \cdot \boldsymbol{n} \;=\; 0, \;\; \boldsymbol{\mathcal{T}} \boldsymbol{n} \cdot \boldsymbol{t} \;=\; 0, & \text{on}\,\Gamma_{g,1} \cup \Gamma_{a,2}, \\ \boldsymbol{\mathcal{T}} \boldsymbol{n} \;=\; \boldsymbol{0}, & \text{on}\,\Gamma_{a,1}, \\ \boldsymbol{u}_* \;=\; \boldsymbol{0}, & \text{on}\,\Gamma_{g,2}, \end{cases} \tag{5.1.1}$$

where

$$\boldsymbol{u}_* = \begin{cases} \boldsymbol{u} - \boldsymbol{u}_p, & \text{on}\,\Gamma_p, \\ \boldsymbol{u}, & \text{elsewhere}, \end{cases} \tag{5.1.2}$$

and

$$\Gamma_{g,1} = \Gamma_s \cap \Omega_g, \tag{5.1.3}$$

$$\Gamma_{g,2} = (\Gamma_r \cup \Gamma_m \cup \Gamma_p) \cap \Omega_g, \tag{5.1.4}$$

$$\Gamma_{a,1} = \Gamma_{r,1} \cap \Omega_a, \tag{5.1.5}$$

$$\Gamma_{a,2} = (\Gamma_m \cup \Gamma_{r,2} \cup \Gamma_p) \cap \Omega_a, \tag{5.1.6}$$

$$\tag{5.1.7}$$

Note that since $\mathrm{tr}(\nabla_{\mathrm{s}} \boldsymbol{u}) = \nabla \cdot \boldsymbol{u} = 0$, the subscript $_d$ can be omitted. Problem (5.1.1) also has a weak formulation. First, define the vector spaces

$$Q := L^2(\Omega), \tag{5.1.8}$$

$$U := \left\{ \boldsymbol{u} \in (H^1(\Omega))^2 \;\middle|\; (\boldsymbol{u}_* \cdot \boldsymbol{n})\Big|_{\Gamma_{g,1} \cup \Gamma_{a,2}} = 0,\; \boldsymbol{u}_*\Big|_{\Gamma_{eg}} = \boldsymbol{0} \right\}, \tag{5.1.9}$$

$$V := \left\{ \boldsymbol{v} \in (H^1(\Omega))^2 \;\middle|\; (\boldsymbol{v} \cdot \boldsymbol{n})\Big|_{\Gamma_{g,1} \cup \Gamma_{a,2}} = 0,\; \boldsymbol{v}\Big|_{\Gamma_{eg}} = \boldsymbol{0} \right\}. \tag{5.1.10}$$

Then, a weak formulation of (5.1.1) is to find $(\boldsymbol{u}, p) \in U \times Q$, such that for all $(\boldsymbol{v}, q) \in V \times Q$,

$$\begin{cases} a(\boldsymbol{v}, \boldsymbol{u}) + b(\boldsymbol{v}, p) = 0, \\ \qquad\qquad b(\boldsymbol{u}, q) = 0. \end{cases} \tag{5.1.11}$$

where $a : \left(H^1(\Omega)\right)^2 \times \left(H^1(\Omega)\right)^2 \mapsto \mathbb{R}$ and $b : \left(H^1(\Omega)\right)^2 \times Q \mapsto \mathbb{R}$ are bilinear forms, defined by

$$a(\boldsymbol{v}, \boldsymbol{u}) = \int_\Omega \left(2\mu \nabla \boldsymbol{v} : \nabla_s \boldsymbol{u}\right) \mathrm{d}\Omega, \tag{5.1.12}$$

$$b(\boldsymbol{v}, q) = \int_\Omega q \nabla \cdot \boldsymbol{v} \mathrm{d}\Omega. \tag{5.1.13}$$

For definitions of the spaces $H^1(\Omega)$ and $L^2(\Omega)$, the reader is referred to Appendix A.3.

5.2 Finite Element Discretisation of Stokes Flow Problems

Variational Stokes flow problem (5.1.11) can be discretised by means of FEMs as follows. First, define finite dimensional vector spaces for the pressure, velocity and weight functions. Let $\{\psi_i\}_{i=1}^M$ and $\{\phi_i\}_{i=1}^N$ be sets of basis functions for the pressure and velocity, respectively. Then, the finite dimensional vector spaces are defined as follows:

$$Q_M := \Big\{\hat{p} = \sum_{i=1}^{M} \psi_i \hat{p}_i \mid \hat{p}_i \in \mathbb{R},\ i = 1, \ldots, M\Big\}, \tag{5.2.1}$$

$$U_N := \Big\{\hat{\boldsymbol{u}} = \sum_{i=1}^{N} \phi_i \hat{\boldsymbol{u}}_i \mid (\hat{\boldsymbol{u}}_* \cdot \boldsymbol{n})\Big|_{\Gamma_{g,1} \cup \Gamma_{a,2}} = 0,\ \hat{\boldsymbol{u}}_*\Big|_{\Gamma_{g,2}} = \boldsymbol{0},\ \hat{\boldsymbol{u}}_i \in \mathbb{R}^2,\ i = 1, \ldots, N\Big\}, \tag{5.2.2}$$

$$V_N := \Big\{\hat{\boldsymbol{v}} = \sum_{i=1}^{N} \phi_i \hat{\boldsymbol{v}}_i \mid (\hat{\boldsymbol{v}} \cdot \boldsymbol{n})\Big|_{\Gamma_{g,1} \cup \Gamma_{a,2}} = 0,\ \hat{\boldsymbol{v}}\Big|_{\Gamma_{g,2}} = \boldsymbol{0},\ \hat{\boldsymbol{v}}_i \in \mathbb{R}^2,\ i = 1, \ldots, N\Big\}, \tag{5.2.3}$$

$$\tag{5.2.4}$$

Next, let $(\hat{\boldsymbol{u}}, \hat{p}) \in U_N \times Q_M$, $(\hat{\boldsymbol{v}}, \hat{p}) \in V_N \times Q_M$. For the sake of notation, introduce the vectors

$$\underset{\sim}{\hat{\boldsymbol{u}}} = \begin{pmatrix} \underset{\sim}{\hat{\boldsymbol{u}}}_1 \\ \underset{\sim}{\hat{\boldsymbol{u}}}_2 \end{pmatrix}, \qquad \underset{\sim}{\hat{\boldsymbol{v}}} = \begin{pmatrix} \underset{\sim}{\hat{\boldsymbol{v}}}_1 \\ \underset{\sim}{\hat{\boldsymbol{v}}}_2 \end{pmatrix}, \qquad \boldsymbol{\phi} = \begin{pmatrix} \phi_1 \\ \vdots \\ \phi_N \end{pmatrix}, \tag{5.2.5}$$

where,

$$\underset{\sim}{\hat{\boldsymbol{u}}}_i = \begin{pmatrix} \hat{u}_{1,i} \\ \vdots \\ \hat{u}_{N,i} \end{pmatrix}, \qquad \underset{\sim}{\hat{\boldsymbol{v}}}_i = \begin{pmatrix} \hat{v}_{1,i} \\ \vdots \\ \hat{v}_{N,i} \end{pmatrix}, \tag{5.2.6}$$

for $i = 1, 2$. In terms of (5.2.5) and (5.2.6), $\hat{\boldsymbol{u}}$ and $\hat{\boldsymbol{v}}$ can be written as

$$\hat{\boldsymbol{u}} = \begin{pmatrix} \boldsymbol{\phi}^T \underset{\sim}{\hat{\boldsymbol{u}}}_1 \\ \boldsymbol{\phi}^T \underset{\sim}{\hat{\boldsymbol{u}}}_2 \end{pmatrix} = \begin{pmatrix} \boldsymbol{\phi}^T & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\phi}^T \end{pmatrix} \underset{\sim}{\hat{\boldsymbol{u}}}, \qquad \hat{\boldsymbol{v}} = \begin{pmatrix} \boldsymbol{\phi}^T \underset{\sim}{\hat{\boldsymbol{v}}}_1 \\ \boldsymbol{\phi}^T \underset{\sim}{\hat{\boldsymbol{v}}}_2 \end{pmatrix} = \begin{pmatrix} \boldsymbol{\phi}^T & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\phi}^T \end{pmatrix} \underset{\sim}{\hat{\boldsymbol{v}}}. \tag{5.2.7}$$

The approximations for p and q in vector notation are

$$\hat{p} = \boldsymbol{\psi}^T \underset{\sim}{\hat{\boldsymbol{p}}}, \qquad \hat{q} = \boldsymbol{\psi}^T \underset{\sim}{\hat{\boldsymbol{q}}}. \tag{5.2.8}$$

where

$$\underset{\sim}{\hat{\boldsymbol{p}}} = \begin{pmatrix} \hat{p}_1 \\ \vdots \\ \hat{p}_M \end{pmatrix}, \qquad \underset{\sim}{\hat{\boldsymbol{q}}} = \begin{pmatrix} \hat{q}_1 \\ \vdots \\ \hat{q}_M \end{pmatrix}. \tag{5.2.9}$$

Substitution of (5.2.7) and (5.2.8) into the double inner product in form (5.1.12) yields

$$\begin{aligned} \nabla \hat{\boldsymbol{v}} : \nabla_s \hat{\boldsymbol{u}} &= \nabla(\boldsymbol{\phi}^T \underset{\sim}{\hat{\boldsymbol{v}}}_1) \cdot \nabla_s(\boldsymbol{\phi}^T \underset{\sim}{\hat{\boldsymbol{u}}}_1) + \nabla(\boldsymbol{\phi}^T \underset{\sim}{\hat{\boldsymbol{v}}}_2) \cdot \nabla_s(\boldsymbol{\phi}^T \underset{\sim}{\hat{\boldsymbol{u}}}_2) \\ &= \underset{\sim}{\hat{\boldsymbol{v}}}_1^T (\nabla \boldsymbol{\phi})^T \nabla_s \boldsymbol{\phi}\, \underset{\sim}{\hat{\boldsymbol{u}}}_1 + \underset{\sim}{\hat{\boldsymbol{v}}}_2^T (\nabla \boldsymbol{\phi})^T \nabla_s \boldsymbol{\phi}\, \underset{\sim}{\hat{\boldsymbol{u}}}_2 \end{aligned} \tag{5.2.10}$$

As a result, the discretised forms can be written as

$$a(\hat{\boldsymbol{v}}, \hat{\boldsymbol{u}}) = \underset{\sim}{\hat{\boldsymbol{v}}}^T \mathcal{K}_{11} \underset{\sim}{\hat{\boldsymbol{u}}}, \tag{5.2.11}$$

$$b(\hat{\boldsymbol{v}}, \hat{q}) = \underset{\sim}{\hat{\boldsymbol{q}}}^T \mathcal{K}_{21} \underset{\sim}{\hat{\boldsymbol{v}}}, \tag{5.2.12}$$

where

$$\mathcal{K}_{11} = \int_\Omega 2\mu \begin{pmatrix} (\nabla \boldsymbol{\phi})^T \nabla_s \boldsymbol{\phi} & \mathbf{0} \\ \mathbf{0} & (\nabla \boldsymbol{\phi})^T \nabla_s \boldsymbol{\phi} \end{pmatrix} \mathrm{d}\Omega, \tag{5.2.13}$$

$$\mathcal{K}_{21} = \int_\Omega \boldsymbol{\psi} \nabla \cdot \begin{pmatrix} \boldsymbol{\phi}^T & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\phi}^T \end{pmatrix} \mathrm{d}\Omega, \tag{5.2.14}$$

The corresponding discretised variational problem is

$$\forall_{\hat{\boldsymbol{v}} \in V_N,\ \hat{q} \in Q_M} \ : \begin{cases} a(\hat{\boldsymbol{v}}, \hat{\boldsymbol{u}}) + b(\hat{\boldsymbol{v}}, \hat{p}) = 0, \\ \qquad\qquad b(\hat{\boldsymbol{u}}, \hat{q}) = 0. \end{cases} \tag{5.2.15}$$

System (5.2.15) holds for any vectors $\underset{\sim}{\hat{\boldsymbol{v}}} \in \mathbb{R}^{2N}$, with $(\boldsymbol{\phi}^T \underset{\sim}{\hat{\boldsymbol{v}}}_1,\ \boldsymbol{\phi}^T \underset{\sim}{\hat{\boldsymbol{v}}}_2)^T \in V_N$, and $\underset{\sim}{\hat{\boldsymbol{q}}} \in \mathbb{R}^M$, so

$$\mathcal{K} \underset{\sim}{\hat{\boldsymbol{\alpha}}} = \mathbf{0}, \tag{5.2.16}$$

where

$$\mathcal{K} = \begin{pmatrix} \mathcal{K}_{11} & \mathcal{K}_{12} \\ \mathcal{K}_{21} & \mathbf{0} \end{pmatrix}, \tag{5.2.17}$$

with $\mathcal{K}_{12} = \mathcal{K}_{21}^T$, and

$$\underset{\sim}{\hat{\boldsymbol{\alpha}}} = \begin{pmatrix} \underset{\sim}{\hat{\boldsymbol{u}}} \\ \underset{\sim}{\hat{\boldsymbol{p}}} \end{pmatrix}. \tag{5.2.18}$$

The discretised Stokes flow problem (5.2.16) forms a system of $2N + M$ equations with $2N + M$ unknowns. However, the discretised continuity equation,

$$\mathcal{K}_{21}\underset{\sim}{\hat{\boldsymbol{u}}} = \mathbf{0}, \tag{5.2.19}$$

forms a system of M equations with $2N$ unknowns. Since it is not desirable that the velocity approximation is completely described by the continuity equation, it should hold that

$$M < 2N. \tag{5.2.20}$$

Indeed, this demand lays some restrictions on the mesh to be used. All elements used in this thesis satisfy this restriction, provided that the mesh is not too coarse.

For simplicity, it is assumed in the following sections that the viscosity μ is constant. For constant viscosity, it holds that

$$\nabla \cdot (2\mu\nabla_s \boldsymbol{u}) \; = \; \mu\Big(\nabla(\nabla \cdot \boldsymbol{u}) + \Delta \boldsymbol{u}\Big) \; = \; \mu\Delta\boldsymbol{u},$$

so matrix $\mathcal{K}_{11}$ can be written as

$$\mathcal{K}_{11} = \mu \int_{\Omega} \begin{pmatrix} (\nabla\boldsymbol{\phi})^T\nabla\boldsymbol{\phi} & \mathbf{0} \\ \mathbf{0} & (\nabla\boldsymbol{\phi})^T\nabla\boldsymbol{\phi} \end{pmatrix} \mathrm{d}\Omega. \tag{5.2.21}$$

In Chapter 7, it is shown that the solver problem for constant viscosity occurs to the same extent as for temperature dependent viscosity.

5.3 Existence and Uniqueness of Solutions

5.3.1 Weak solutions

Existence and uniqueness of solutions of (5.1.11) is stated by the following theorem [12, 5]:

Theorem 5.3.1. *Let a be a symmetric and positive semi-definite bilinear form on $V \times V$. Then, variational problem (5.1.11) has a unique solution, iff a satisfies*

$$\exists_{\alpha>0} \, \forall_{\boldsymbol{v}\in V(0)} \; : \; a(\boldsymbol{v},\boldsymbol{v}) \geq \alpha \, \|\boldsymbol{v}\|_V^2, \tag{5.3.1}$$

where

$$V(0) = \Big\{ \boldsymbol{v} \in V \, \Big| \, \forall_{q\in Q} \; : \; b(\boldsymbol{v},q) = 0 \Big\}, \tag{5.3.2}$$

and b satisfies

$$\exists_{\beta>0} \; : \; \inf_{q\in Q} \sup_{\boldsymbol{v}\in V,\, \boldsymbol{v}\neq \boldsymbol{0}} \frac{b(\boldsymbol{v},q)}{\|\boldsymbol{v}\|_V} \geq \beta \, \|q\|_Q. \tag{5.3.3}$$

The proof of theorem 5.3.1 can be found in [5]. Condition (5.3.1) is referred to as the ellipticity condition and condition (5.3.3) is called the inf-sup condition or Brezzi-Babuška condition. Clearly, for constant viscosity, the form a is symmetric and positive semi-definite. The presence of Dirichlet BCs ensures that the ellipticity condition is satisfied [2], Ch. 3. The inf-sup condition is related to the shape of Ω and is satisfied for fairly complicated flow domains [12], Ch. 8.

5.3.2 Finite Element Solutions

Similar conditions as (5.3.1) and (5.3.3) exist for the discrete Stokes flow problem (5.2.16). Introduce the discrete ellipticity condition,

$$\exists_{\hat{\alpha}>0} \, \forall_{\hat{\boldsymbol{v}}\in V_N(0)} \; : \; a(\hat{\boldsymbol{v}},\hat{\boldsymbol{v}}) \geq \hat{\alpha} \, \|\hat{\boldsymbol{v}}\|_V^2, \tag{5.3.4}$$

and the discrete inf-sup condition,

$$\exists_{\hat{\beta}>0} \; : \; \inf_{\hat{q}\in Q_M} \sup_{\hat{\boldsymbol{v}}\in V_N, \hat{\boldsymbol{v}}\neq \boldsymbol{0}} \frac{b(\hat{\boldsymbol{v}},\hat{q})}{\|\hat{\boldsymbol{v}}\|_V} \geq \hat{\beta} \, \|\hat{q}\|_Q, \tag{5.3.5}$$

where $\hat{\alpha}$ and $\hat{\beta}$ do not depend on the size of the elements. Here

$$V_N(0) = \Big\{ \hat{\boldsymbol{v}} \in V_N \, \Big| \, \forall_{\hat{q}\in Q_M} \; : \; b(\hat{\boldsymbol{v}},\hat{q}) = 0 \Big\}. \tag{5.3.6}$$

It can be proven that if the discrete inf-sup condition is satisfied, then matrix $\mathcal{K}_{21}$ has full rank [12], Ch. 8. Since condition (5.3.4) implies that $\mathcal{K}_{11}$ is positive definite, it follows that $\mathcal{K}$ is non-singular if both (5.3.4) and (5.3.5) are satisfied. If $V_N \subset V$, then condition (5.3.4) is automatically satisfied. Satisfaction of condition (5.3.5) depends on the mesh and the type of element used. Elements that satisfy condition (5.3.5) are called stable elements. Stable elements for Stokes flow problem (5.1.1) are discussed in section 5.5.

5.4 Solution Methods

Finite element discretisations of the Stokes flow equations are generally known as one of the most prominent sources of indefinite linear systems [12], Ch. 8. They are characterised by a large zero-block on the diagonal in the right-bottom of the stiffness matrix, which may lead to complications in solving the system. The literature consulted supplies three different relevant methods to deal with this kind of problem.

Penalty Function Methods

The continuity equation is perturbed by a small factor ε of the pressure:

$$\nabla \cdot \boldsymbol{u} + \varepsilon p = 0. \tag{5.4.1}$$

It can be proven that the solution of the perturbed system approximates the original solution, provided that ε is small enough [9], Ch. 8. Advantages of the penalty function method are [32, 30, 9]:

- the system of equations is reduced, as the linear system for the continuity equation can be decoupled from the linear system for the momentum equation,
- the calculation time decreases, as no partial pivoting is required.

On the other hand, the penalty method also has some disadvantages [32, 30]:

- the choice of ε depends on the magnitude of the unknown pressure,
- the inverse parameter $\frac{1}{\varepsilon}$ outweighs other entries in the stiffness matrix, causing a large condition number.

A consequence of the latter disadvantage is that the penalty function is not suitable for large-scaled problems. Since the condition number of the stiffness matrix of the discrete Stokes flow problem in the pressing model is already substantial, as mentioned in Chapter 1, the penalty function method is not a preferable choice [32, 30, 9].

Methods of Divergence-Free Basis Functions

The basis functions are chosen, such that the continuity equation is automatically satisfied. So, the vector spaces of velocity basis functions, U_N and V_N, are extended with additional incompressibility conditions,

$$\nabla \cdot \hat{\boldsymbol{u}} = 0, \text{ resp. } \nabla \cdot \hat{\boldsymbol{v}} = 0.$$

An advantage is that the number of unknowns is strongly reduced. However, the method can only be applied for one specific type of elements: the so-called Crouzeix-Raviart elements. Furthermore, some complications may arise in the application of this method. Basis functions that satisfy the incompressibility condition have to be constructed, boundary conditions must be transformed and the solution is to be transformed back to its original degrees of freedom. The method is not (yet) available in Sepran, and is therefore not used in the pressing model. For more information, the reader is referred to C. Cuvelier Et Al [9], Ch. 9.

Integrated Methods

The integrated method does not uncouple the velocity and pressure unknowns, which is the case in the penalty function method and the method of divergence-free basis functions. Often, partial pivoting is applied to deal with the zero diagonal elements. However, Rehman et al [26] state that this technique is not used in Sepran, as it can change the profile or bandwidth of the coefficient matrix. Instead, the unknowns are reordered, such that the velocity unknowns are calculated before the pressure unknowns in the same nodes. Ordering methods are discussed in section 6.2.1. A disadvantage of integrated methods is that the matrix, and hence also the calculation time, become much larger [30]. This method is used in TNO Glass Group's pressing model.

5.5 Stable Elements

The purpose of this section is to discuss which stable elements are suitable for application in the pressing model. Section 5.5.1 introduces so-called Mini-elements, which are used in TNO Glass Group's pressing model. Section 5.5.2 discusses so-called Taylor-Hood elements as possible alternatives. Unfortunately, in Chapter 7 it appears that the solver performance for Taylor-Hood elements is not generally much better than for Mini-elements. Finally, section 5.5.3 briefly discusses why other elements available in Sepran are rejected in this thesis in favour of the aforementioned elements.

5.5.1 Mini-Elements

Mini-elements use linear basis functions in joint nodes for both velocity end pressure in the vertices, assigning the same basis functions to pressure and velocity unknowns. These properties in themselves do not suffice for an element to be stable. For example, R. Pierre [23] shows in a simple example that triangular Courant elements (linear velocity - linear pressure elements) do not have a unique pressure solution (such solutions are said to contain spurious pressure modes). Therefore, the vector space of velocity basis functions for the Courant element is enriched by adding a so-called bubble function in an additional node in the barycentre of the element. This bubble function is nothing more than a Lagrange interpolant involving the nodes in the vertices of the element, which is just the normalised product of the linear basis functions in these nodes:

$$b_e(\boldsymbol{x}) = \prod_i \frac{\phi_{e,i}(\boldsymbol{x})}{\phi_{e,i}(\hat{\boldsymbol{x}}_0)}, \qquad \boldsymbol{x} \in e \tag{5.5.1}$$

with $\hat{\boldsymbol{x}}_0$ the position of the barycentre and indices i corresponding to the vertices $\hat{\boldsymbol{x}}_i$ of the element, $i = 1, \ldots l_e$. The bubble function can be seen as an additional velocity basis function $\phi_{e,0}$ (see Figure 5.1a and Figure 5.1b). V. Girault and P. A. Raviart [14], Ch. II, have shown that triangular Mini-elements are stable for uniform Stokes flow problems. However, R. Pierre [23] states that in many tests, the pressure solutions is plagued with small amplitude oscillations. In order to reduce these, the mesh has to be refined, which leads to an increase in computational time, especially in view of the solver problem. In section 5.6, it can be seen that for Mini-elements a stabilised formulation of the Stokes flow problem can be derived. For results for non-uniform Stokes flow problems, the reader is referred to Chapter 7.

5.5.2 Taylor-Hood Elements

Taylor-Hood elements use joint nodes for the pressure and the velocity unknowns. They use k^{th} order continuous piecewise polynomials for the velocity and $(k-1)^{\text{th}}$ order continuous piecewise polynomials for the pressure, for some integer $k > 1$. Since Taylor-Hood elements with joint boundaries

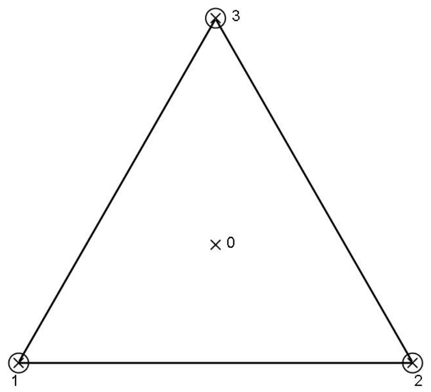


Figure 5.1a: Linear Mini-element ($P_1 - P_1$/bubble). The element uses 4 nodes for the velocity unknowns (x), including the bubble function in the barycentre, and 3 nodes for the pressure nodes (O).

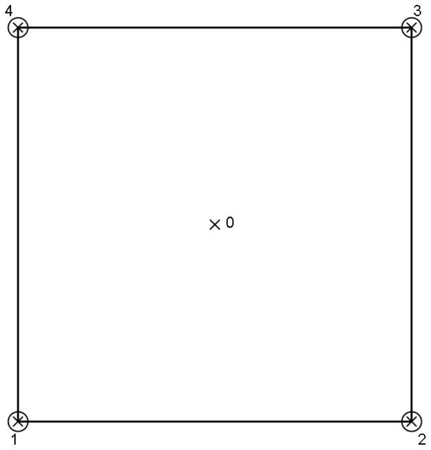


Figure 5.1b: Bilinear Mini-element ($Q_1 - Q_1$/bubble). The element uses 5 nodes for the velocity unknowns (x), including the bubble function in the barycentre, and 4 nodes for the pressure nodes (O).

use the same approximations for the velocity and the pressure, both the velocity and the pressure are continuous in Ω. In this thesis, only Taylor-Hood elements with $k = 2$ are considered.

V. Girault and P. A. Raviart [14], Ch. II, have shown that Taylor-Hood elements are stable for uniform Stokes flow problems. In addition, F. Brezzi and R. S. Falk [6] prove stability of higher-order Taylor-Hood elements, also for uniform Stokes flow problems. In Chapter 7, some results for Taylor-Hood elements for a test model for the simulation of a pressing time step are given.

Figure 5.2a shows a triangular Taylor-Hood element with quadratic velocity and linear pressure ($P_2 - P_1$) and Figure 5.2b shows a quadrilateral Taylor-Hood element with biquadratic velocity and bilinear pressure ($Q_2 - Q_1$). Let h denote the maximum of the edge lengths of the Taylor-Hood element. Then the accuracy of the velocity approximations is $O(h^3)$ and the accuracy of the pressure approximations is $O(h^2)$ [9], Ch. 7. C. Cuvelier et al [9] show how the basis functions for Taylor-Hood elements can be constructed.

5.5.3 Elements for the Integrated Method

Basically, in Sepran three classes of elements can be used for the integrated method, namely Mini-elements, Taylor-Hood elements and so-called Crouzeix-Raviart elements. The first two element types are fairly suitable for the integrated method, especially Mini-elements, because of their stabilisation property (see section 5.6). The third element type is often used in penalty function methods and methods of divergence-free basis functions.

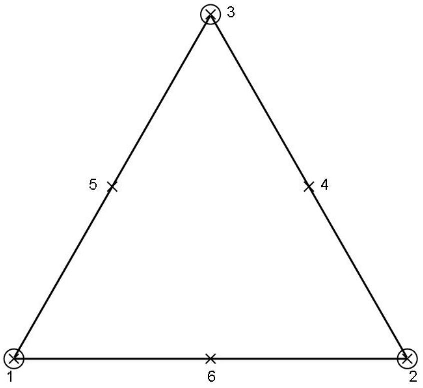


Figure 5.2a: Quadratic velocity - linear pressure Taylor-Hood element ($P_2 - P_1$). The element uses 6 nodes for the velocity unknowns (x) and 3 nodes for the pressure nodes (O).

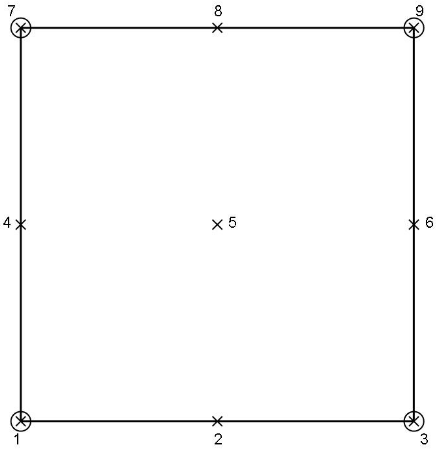


Figure 5.2b: Biquadratic velocity - bilinear pressure Taylor-Hood element ($Q_2 - Q_1$). The element uses 9 nodes for the velocity unknowns (x) and 4 nodes for the pressure nodes (O).

Crouzeix-Raviart elements use boundary nodes for the velocity unknowns and interior nodes for the pressure unknowns. As a result, the pressure approximation is discontinuous in Ω. Sometimes, additional joint interior nodes for both velocity and pressure unknowns are used.

For an evaluation of Crouzeix-Raviart elements for the integrated method, it is convenient to compare them with Taylor-Hood elements with basis functions of the same degrees. For integrated methods, Taylor-Hood elements are slightly computationally cheaper than Crouziex-Raviart elements, since the amount of unknowns is somewhat smaller. Furthermore, Crouziex-Raviart elements are often not accurate enough for axi-symmetric problems. An advantage of Crouziex-Raviart elements is that the continuity equation is satisfied at element level, whereas Taylor-Hood elements are only mass-conservative on the entire domain Ω. However, it is not clear whether the accuracy of the solution is influenced by this property. Overall, Taylor-Hood elements are preferred to Crouziex-Raviart elements in this thesis, the latter elements not being used [9], Ch. 7.

5.6 Stabilised Stokes Flow Problems for Triangular Mini-Elements

In TNO Glass Group's pressing model, the bubble functions in the triangular Mini-elements are eliminated at element level by means of so-called static condensation. In static condensation methods, interior and boundary degrees of freedom are distinguished, i.e. degrees of freedom corresponding to interior and boundary nodes of the elements, respectively. For the restriction of system (5.2.16) to element level, this gives

$$\begin{pmatrix} \mathcal{K}_e^{(bb)} & \mathcal{K}_e^{(ib)} \\ \mathcal{K}_e^{(bi)} & \mathcal{K}_e^{(ii)} \end{pmatrix} \begin{pmatrix} \underset{\sim}{\hat{\alpha}}_e^{(b)} \\ \underset{\sim}{\hat{\alpha}}_e^{(i)} \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}_e \\ \mathbf{0} \end{pmatrix}, \tag{5.6.1}$$

where the superscripts (b) and (i) refer to boundary unknowns and interior unknowns, respectively. Here,

$$\sum_e \boldsymbol{f}_e = \mathbf{0}, \tag{5.6.2}$$

so that global system (5.2.16) holds. Elimination of the interior unknowns yields

$$\widetilde{\mathcal{K}}_e \underset{\sim}{\hat{\alpha}}_e^{(b)} = \boldsymbol{f}_e, \tag{5.6.3}$$

where

$$\widetilde{\mathcal{K}}_e = \mathcal{K}_e^{(bb)} - \mathcal{K}_e^{(bi)} \left(\mathcal{K}_e^{(ii)}\right)^{-1} \mathcal{K}_e^{(ib)}. \tag{5.6.4}$$

In this section, it is shown that static condensation at element level for Mini-elements is the same as stabilising the Stokes flow equations for Courant elements. This has already been verified by R. Pierre [23] and J. E. Maitre and E. Wabo [18] for static condensation applied to the global system of Stokes flow equations. Although in theory the principle is the same for both global and element-wise static condensation methods, experimental results in Sepran show that the solver performance is slightly better if static condensation at element level is applied (see Chapter 7). Here, it should be mentioned that it is not clear whether Sepran applies static condensation on the global system in case no static condensation at element level is applied. Main sources for this section are [23] and [18].

The internal nodes of the Mini-elements, corresponding to the bubble functions, are eliminated by means of (5.6.3)-(5.6.4). For simplicity, consider a linear Mini-element e. Let $\{\varphi_1, \varphi_2, \varphi_3\}$ be the set of basis functions corresponding to the boundary nodes and let $|e|$ denote the measure of e. First, distinguish solution vectors $\underset{\sim}{\hat{\alpha}}_e^{(b)}$, comprising the 9 boundary unknowns, and $\underset{\sim}{\hat{\alpha}}_e^{(i)}$, comprising the 2 interior velocity unknowns. The coefficient matrices corresponding to the boundary and interior unknowns are

$$\mathcal{K}_e^{(bb)} = \begin{pmatrix} \mathcal{K}_{e,11}^{(bb)} & \mathcal{K}_{e,12}^{(bb)} \\ \mathcal{K}_{e,21}^{(bb)} & \mathbf{0} \end{pmatrix}, \tag{5.6.5}$$

$$\mathcal{K}_e^{(ib)} = \begin{pmatrix} \mathcal{K}_{e,11}^{(ib)} & \mathcal{K}_{e,12}^{(ib)} \end{pmatrix}, \tag{5.6.6}$$

$$\mathcal{K}_e^{(bi)} = \begin{pmatrix} \mathcal{K}_{e,11}^{(bi)} \\ \mathcal{K}_{e,21}^{(bi)} \end{pmatrix}, \tag{5.6.7}$$

$$\mathcal{K}_e^{(ii)} = \begin{pmatrix} \mathcal{K}_{e,11}^{(ii)} \end{pmatrix}. \tag{5.6.8}$$

Here, $\mathcal{K}_e^{(bb)}$ is the coefficient matrices for the corresponding linear Courant element. The second and third coefficient matrices are each others transposes, $\mathcal{K}_e^{(ib)} = \left(\mathcal{K}_e^{(bi)}\right)^T$. Substitution of (5.6.5)-(5.6.8) into (5.6.4) gives

$$\widetilde{\mathcal{K}}_e = \begin{pmatrix} \mathcal{K}_{e,11}^{(bb)} - \mathcal{K}_{e,11}^{(bi)}\left(\mathcal{K}_{e,11}^{(ii)}\right)^{-1}\mathcal{K}_{e,11}^{(ib)} & \mathcal{K}_{e,12}^{(bb)} - \mathcal{K}_{e,11}^{(bi)}\left(\mathcal{K}_{e,11}^{(ii)}\right)^{-1}\mathcal{K}_{e,12}^{(ib)} \\ \mathcal{K}_{e,21}^{(bb)} - \mathcal{K}_{e,21}^{(bi)}\left(\mathcal{K}_{e,11}^{(ii)}\right)^{-1}\mathcal{K}_{e,11}^{(ib)} & -\mathcal{K}_{e,21}^{(bi)}\left(\mathcal{K}_{e,11}^{(ii)}\right)^{-1}\mathcal{K}_{e,12}^{(ib)} \end{pmatrix}. \tag{5.6.9}$$

The matrices in (5.6.9) can be evaluated as follows. For the matrix $\mathcal{K}_{e,11}^{(bi)}$, it holds that

$$\begin{aligned} \mathcal{K}_{e,11}^{(bi)} &= \mu \int_e \begin{pmatrix} (\nabla\boldsymbol{\varphi})^T \nabla b_e & \mathbf{0} \\ \mathbf{0} & (\nabla\boldsymbol{\varphi})^T \nabla b_e \end{pmatrix} \mathrm{d}\Omega \\ &= \mu \begin{pmatrix} \nabla\boldsymbol{\varphi} & \mathbf{0} \\ \mathbf{0} & \nabla\boldsymbol{\varphi} \end{pmatrix}^T \int_{\partial e} \begin{pmatrix} b_e \cdot \boldsymbol{n} & \mathbf{0} \\ \mathbf{0} & b_e \cdot \boldsymbol{n} \end{pmatrix} \mathrm{d}\Gamma \\ &= \mathbf{0}. \end{aligned} \tag{5.6.10}$$

Hence, only the matrices corresponding to the boundary nodes and the right-bottom matrix in (5.6.9) remain:

$$\widetilde{\mathcal{K}}_e = \begin{pmatrix} \mathcal{K}_{e,11}^{(bb)} & \mathcal{K}_{e,12}^{(bb)} \\ \mathcal{K}_{e,21}^{(bb)} & \widetilde{\mathcal{K}}_{e,22} \end{pmatrix}, \tag{5.6.11}$$

where

$$\widetilde{\mathcal{K}}_{e,22} = -\mathcal{K}_{e,21}^{(bi)}\left(\mathcal{K}_{e,11}^{(ii)}\right)^{-1}\mathcal{K}_{e,12}^{(ib)}. \tag{5.6.12}$$

Matrix $\mathcal{K}^{(ii)}_{e,11}$ is a 2×2 diagonal matrix,

$$\mathcal{K}^{(ii)}_{e,11} \;=\; \mu \int_e \nabla b_e \cdot \nabla b_e \mathrm{d}\Omega \, \mathcal{I}_2 \;=\; \mu \, \|\nabla b_e\|_{L^2(e)} \mathcal{I}_2 \tag{5.6.13}$$

So, the right-bottom matrix $\widetilde{\mathcal{K}}_{e,22}$ can be written as

$$\begin{aligned} \widetilde{\mathcal{K}}_{e,22} &= -(\mu \, \|\nabla b_e\|_{L^2(e)})^{-1} \int_e \boldsymbol{\varphi}(\nabla b_e)^T \mathrm{d}\Omega \int_e (\nabla b_e)\boldsymbol{\varphi}^T \mathrm{d}\Omega \\ &= (\mu \, \|\nabla b_e\|_{L^2(e)})^{-1} \int_e (\nabla \boldsymbol{\varphi})^T b_e \mathrm{d}\Omega \int_e b_e \nabla \boldsymbol{\varphi} \mathrm{d}\Omega \\ &= \kappa_e \int_e (\nabla \boldsymbol{\varphi})^T (\nabla \boldsymbol{\varphi}) b_e \mathrm{d}\Omega \end{aligned} \tag{5.6.14}$$

where in the second equation Gauss' gradient theorem is applied, and $\kappa_e > 0$ is a constant proportional to $|e|$, given by

$$\kappa_e = \frac{\int_e b_e \mathrm{d}\Omega}{\mu \, \|\nabla b_e\|_{L^2(e)}}. \tag{5.6.15}$$

Finally, the global system of stabilised Stokes flow equations for triangular Mini-elements is

$$\widetilde{\mathcal{K}} \underset{\sim}{\hat{\alpha}}^{(b)} = \mathbf{0}, \tag{5.6.16}$$

where

$$\widetilde{\mathcal{K}} = \begin{pmatrix} \widetilde{\mathcal{K}}_{11} & \widetilde{\mathcal{K}}_{12} \\ \widetilde{\mathcal{K}}_{21} & \widetilde{\mathcal{K}}_{22} \end{pmatrix}, \tag{5.6.17}$$

Here, the matrices $\widetilde{\mathcal{K}}_{11}$, $\widetilde{\mathcal{K}}_{12}$ and $\widetilde{\mathcal{K}}_{21}$ correspond to the boundary nodes of the Mini-elements and hence they equal the matrices $\mathcal{K}_{11}$, $\mathcal{K}_{12}$ and $\mathcal{K}_{21}$ for the corresponding Courant elements. Furthermore, the right-bottom matrix $\widetilde{\mathcal{K}}_{22}$ is the assembly of the contributions of all elements:

$$\widetilde{\mathcal{K}}_{22} = \sum_e \kappa_e \boldsymbol{\mathcal{P}}_{e,22} \Big(\int_e (\nabla \boldsymbol{\varphi})^T (\nabla \boldsymbol{\varphi}) b_e \mathrm{d}\Omega \Big) \boldsymbol{\mathcal{P}}^T_{e,22}, \tag{5.6.18}$$

where $\boldsymbol{\mathcal{P}}_{e,22}$ is a $N \times 3$ projection matrix that maps the entries of the 3×3 matrix $\widetilde{\mathcal{K}}_{e,22}$ onto the entries of the $N \times N$ matrix $\widetilde{\mathcal{K}}_{22}$. So, system (5.6.16) corresponds to a stabilised variational Stokes flow problem for linear Courant elements: find $(\boldsymbol{u}, p) \in U \times Q$, such that for all $(\boldsymbol{v}, p) \in V \times Q$,

$$\begin{cases} a(\boldsymbol{v}, \boldsymbol{u}) + b(\boldsymbol{v}, p) = 0, \\ b(\boldsymbol{u}, q) + c(p, q) = 0. \end{cases} \tag{5.6.19}$$

where $c : Q \times Q \mapsto \mathbb{R}$ is a bilinear form, given by

$$c(p, q) = \sum_e \kappa_e \int_e (\nabla q \cdot \nabla p) b_e \mathrm{d}\Omega. \tag{5.6.20}$$

J. E. Maitre and E. Wabo [18] prove that such a stabilised Stokes flow problem satisfies the inf-sup condition.

5.7 Construction of a Mesh in TNO Glass Group's Axi-Symmetrical Pressing Model

TNO Glass Group's axi-symmetrical pressing model[1] uses a mesh consisting of triangular Mini-elements. This mesh is constructed for the final plunger position. Figure 5.3a- 5.3d show pictures of the deformation of a coarse mesh in the pressing model from $t = 0$ s to $t = 1.5$ s. The pressing time corresponding to the figures is exactly $t = 1.5$ s, so the mesh in Figure 5.3d is the one provided as input for the pressing model. After the mesh for the lowest plunger position is constructed, it is stretched in vertical direction by raising the plunger to its initial position. In each subsequent time step, a new finite element mesh is generated that fits the new plunger position. Each mesh consists of the same numbers of elements along each dimension of the domain. So, the mesh is compressed by the plunger, just as the flow domain, until the mesh for the final plunger position is again obtained. A disadvantage of this method is that the discrete Stokes flow problem can become ill-conditioned as the triangles are flattened or stretched. This matter is discussed in Chapter 7.

[1] That is, TNO Glass Group's pressing model in which the option ”axi-symmetric” is switched on (see section 3.5)

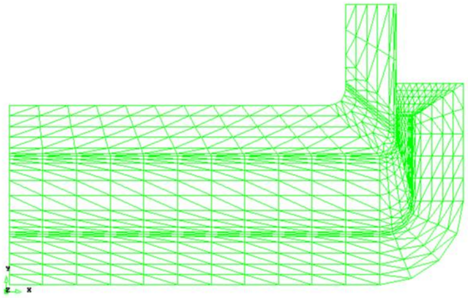


Figure 5.3a: Coarse mesh in TNO Glass Group's axi-symmetrical pressing model at *t* = 0 s.

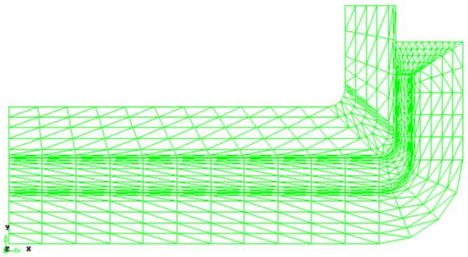


Figure 5.3b: Coarse mesh in TNO Glass Group's axi-symmetrical pressing model at *t* = 0.5 s.

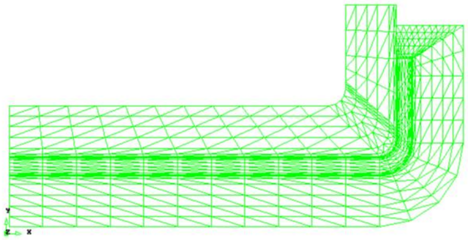


Figure 5.3c: Coarse mesh in TNO Glass Group's axi-symmetrical pressing model at *t* = 1.0 s.

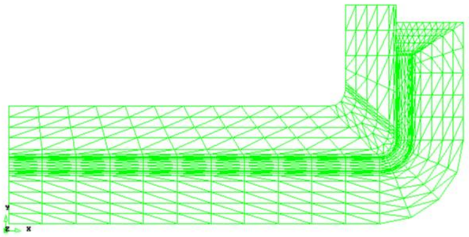


Figure 5.3d: Coarse mesh in TNO Glass Group's axi-symmetrical pressing model at *t* = 1.5 s.

5.8 Conclusions of Finite Element Analysis

In conclusion, the discretisation methods for Stokes flow problems in TNO Glass Group's Axi-Symmetrical Pressing Model are stable and produce well-defined systems of equations. From this point of view, no disproportionate problems should be expected for stable linear solvers, such as direct methods. On the other hand, less robust linear solvers, such as most iterative solvers, may terminate as the discrete Stokes flow problem becomes larger. So, although the choice of the discretisation methods in this chapter is validated, a robust solution method for the resulting system of equations is absolutely essential. Chapter 6 deals with preconditioners and iterative solvers for discrete Stokes flow problems.

Chapter 6

Linear Solvers and Preconditioners

The solver problem discussed in section 1.3 reveals the importance of a robust linear solver for Stokes flow problems. B. Fischer [12] states that it is precisely the class of Stokes flow problems which motivated the search for stable and efficient solvers for symmetric, indefinite systems. This chapter discusses which linear solvers and preconditioners are particularly suitable for Stokes flow problems and analyses their performance. Here, solvers and preconditioners that are available in TNO Glass Group's pressing model are highlighted. Section 6.1 gives a general analysis of the performance of linear solvers for FEMS. Here the performance of iterative solvers is emphasised, but also other linear solvers for Stokes flow problems are briefly discussed. Section 6.2 deals with ILU preconditioning, which is applied in TNO Glass Group's pressing model. In this section, also methods for reordering the unknowns and additional fill-in for ILU are discussed. Finally, section 6.3 suggest some other preconditioning techniques for Stokes flow problems.

6.1 Analysis of the Linear Solver Performance for Linear Systems Arising from Finite Element Methods

In section 3.5, some iterative solvers for non-symmetric matrices are mentioned. Among these, GMRES [28] and BiCGstab [37] belong to the most popular ones. For non-symmetric matrices, BiCGstab generally provides much better results than other conjugate gradient-like methods, such as CGS and BiCG [37]. Furthermore, in many tests the performance of BiCGstab or GMRES is much better than for most minimisation methods, such as QMR [28]. In Chapter 7, the performance of GMRES for TNO Glass Group's pressing model is briefly discussed. In addition, U. M. Yang [39] also suggests the CG method on normal equations, but since the condition number of the matrix for the discrete Stokes flow problem is considerably large, this is not an obvious choice. In this thesis, the majority of the tests have been carried out by means of a BiCGstab solver.

After observing the solver problem in section 1.3, the questions may arise how many iterations and which increase in iterations for mesh refinement is reasonable. In fact, for many iterative solvers, upper bounds for the numbers of iterations are known. For example, for CG it can be proven that [35, 2]

$$N < O\Big(\sqrt{\kappa(\mathcal{K})} \ \log\Big(\frac{1}{\epsilon}\Big)\Big). \tag{6.1.1}$$

Different orders of magnitudes of the required number of iterations for different iterative solvers are given in Table 6.1, which is presented in [19], Ch. 26.9. From Table 6.1, it can be seen that an upper bound as in (6.1.1) also holds for other conjugate gradient-like solvers, such as BiCGstab, and for many minimisation methods, such as GMRES. In addition, O. Axelson and V. A. Barker [2] state that for Galerkin FEMs,

$$\mathcal{K} \sim O(h^{-2}), \tag{6.1.2}$$

where h is the largest edge length in the finite element mesh. Since the condition number in TNO Glass Group's pressing model is observed to be substantial ($O(10^{22})$ for coarse meshes), the number of BiCGstab iterations can reasonably become huge. So, upper bound (6.1.1) would simply mean that the condition number should be reduced in order to guarantee convergence within an acceptable number of iterations. However, in Chapter 7 it is shown that divergence also occurs for condition numbers of the order of magnitude $O(10^4)$. Here, divergence means that the residual grows out of bounds, so that the iterative process is terminated. Apparently, some other aspects are playing a role, so that the upper bound is not properly satisfied for the non-symmetric iterative solvers that were tested (e.g. GMRES and BiCGstab), even though their performance is good in general.

Table 6.1: Iterative methods with corresponding conditions for the coefficient matrix and required number of iterations.

Type	Methods	Conditions	# iterations
static	Jacobi, SSOR, ...	non-trivial	$O(\kappa(\mathcal{K}))$
minimisation	GMRES, QMR, ...	symmetric part positive definite	$\sqrt{O(\kappa(\mathcal{K}))}$
conjugate gradient-like	BiCGstab, CGS, BiCG, ...	non-trivial/none	$\sqrt{O(\kappa(\mathcal{K}))}$
optimal	Multigrid, ...	major non-trivial	1

A type of optimal methods for which there is a growing interest, particularly for application to Stokes flow problems, is the class of so-called multigrid methods. Kan Et Al [15] state that multigrid methods have become one of the most successful iterative techniques over the past 30 years. Multigrid methods accelerate the solver process by interpolation of information on the solution from a coarse mesh onto a fine mesh. In comparison with other methods, multigrid methods depend much less on the structure of the equations and the BCs. Moreover, the number of computations increases linearly with the number of unknowns, although C. C. Douglas [10] states that this may be different for rather complicated problems. According to C. C. Douglas [10], multigrid methods are difficult to apply to nontrivial problems, although it is usually worth the effort. Furthermore, implementing multigrid methods on unstructured meshes leads to some serious data-structure issues. A multigrid method can as well be used as a preconditioner for BiCGstab; typically one multi-grid cycle is used as a preconditioner for a conjugate gradient-like solver. Unfortunately, they are not (yet) supported in Sepran [10, 1, 15].

6.2 Incomplete LU Factorisation Preconditioners

The efficiency and robustness of iterative solvers can be improved by using preconditioning. In general, the reliability of iterative solvers highly depends on the quality of the preconditioner used. Since all iterative solvers applied reveal lack of robustness for the Stokes flow problem in TNO Glass Group's pressing model, the choice of a suitable preconditioner may be crucial in order to improve the iterative solver performance. In this section, the preconditioner currently used in TNO Glass Group's pressing model is analysed. Section 6.3 discusses some preconditioners that are particularly suitable for Stokes flow problems.

In TNO Glass Group's pressing model, incomplete LU factorisation (ILU) preconditioning is used. The ILU preconditioner is known to perform quite well for a wide range of practical application. An advantage of the ILU preconditioner is that it is rather easy and inexpensive to compute. Disadvantages are that ILU preconditioning often leads to a crude approximation, so that many iterations are required for convergence, and that the convergence rate is quite sensitive with respect to the ordering of the unknowns. In section 7.2, results for the ILU preconditioner are compared with results for some other preconditioners in Sepran [28, 22].

Incomplete LU factorisation computes a sparse lower triangular matrix $\boldsymbol{L}$ and a sparse upper triangular matrix $\boldsymbol{U}$, such that certain entries are zero in the residual matrix, given by

$$\mathcal{R} = \boldsymbol{L}\boldsymbol{U} - \mathcal{K}. \tag{6.2.1}$$

The factorisation is the result of the application of Gaussian elimination to a certain pattern P in the coefficient matrix $\mathcal{K}$ that contains the sparsity pattern. ILU can be proven to be feasible if $\mathcal{K}$ is an M-matrix [28]. However, even if the coefficient matrix is symmetric positive definite, the ILU preconditioner is not positive definite, so that efficient algorithms such as MINRES and SYMMLQ [12] cannot be applied [26].

6.2.1 Ordering Methods

Since the ILU preconditioner uses Gaussian elimination, additional care should be taken of zero entries on the diagonal. Zero diagonal entries occur in the discrete Stokes flow problem with the integrated solution method. Because in Sepran no pivoting is applied, it is necessary to reorder the unknowns in order to prevent the presence of zero diagonal entries in the Gaussian elimination process.

Prevention of zero diagonal entries in the Gaussian elimination process is not the only reason for the application of ordering methods. M. Benzi et al [4] state that if either $\|\mathcal{R}\|_F$ or $\|\mathcal{R}(\boldsymbol{LU})^{-1}\|_F$ is large, where $\|\ \cdot\ \|_F$ is the Frobenius norm, the preconditioned iterative solver may fail to converge. One way to overcome this phenomenon is to allow more fill-in (non-zero entries) by the Gaussian elimination process. This is discussed in section 6.2.2. On the other hand, it might be sufficient to reorder the unknowns, before applying ILU. M. Benzi et al [4] state that although the original purpose of ordering methods was improvement of direct solvers, they can also have a favourable effect on the performance of preconditioned iterative solvers applied to nonsymmetric linear systems [4].

In general, methods for reordering unknowns attempt to reduce the number of zero entries in the coefficient matrix that are adjusted by the Gaussian elimination process. Y. Saad [28] gives the following simple example.

Example 6.2.1. *Consider the linear system*

$$\begin{pmatrix} a_{11} & 0 & a_{13} & 0 \\ 0 & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & 0 \\ 0 & a_{42} & 0 & a_{44} \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{pmatrix}. \tag{6.2.2}$$

Here, the sparsity pattern is contained in the main diagonal and four sub-diagonals. So for ILU, it is natural to choose the pattern

$$P = \left\{ a_{ij} \;\middle|\; |i-j| \leq 2 \right\}.$$

Alternatively, reorder the unknowns by interchanging x_2 and x_3. The resulting system is

$$\begin{pmatrix} a_{11} & a_{13} & 0 & 0 \\ a_{31} & a_{3} & a_{32} & 0 \\ 0 & a_{23} & a_{22} & a_{24} \\ 0 & 0 & a_{42} & a_{44} \end{pmatrix}\begin{pmatrix} x_1 \\ x_3 \\ x_2 \\ x_4 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_3 \\ b_2 \\ b_4 \end{pmatrix}. \tag{6.2.3}$$

Obviously, the new sparsity pattern consists of the main diagonal and only two sub-diagonals, which form a smaller pattern P' for the Gaussian elimination process. Note that pattern P has 4 more zero entries than pattern P'. These zero entries are adjusted by the Gaussian elimination process. As a result, the ILU of (6.2.2) will contain more fill-in, i.e. non-zero entries, than the ILU of (6.2.3).

A class of ordering methods that is particularly useful for iterative solvers is the class of so-called level-set ordering methods. A level set is defined recursively as the set of all unmarked neighbours of all nodes of a previous level set. Available level-set ordering methods in Sepran are the Sloan algorithm and the Cuthill Mc Kee algorithm [26]. In TNO Glass Group's pressing model, the Sloan algorithm is applied. Both methods attempt to reduce the profile of the matrix. In literature also the reverse Cuthill Mc Kee algorithm is suggested, in which the Cuthill Mc Kee ordering of the unknowns is reordered backward. M. Benzi Et Al [4] conclude from a wide range of applications that reverse Cuthill Mc Kee appears to be superior to the regular Cuthill Mc Kee orderings in the context of incomplete factorisation preconditioning. Y. Saad [28] gives the following explanation for this. In the case of regular Cuthill Mc Kee ordering, small arrows on the diagonal pointing upward can be observed in the matrix sparsity pattern (see Figure 6.1). For such patterns, Gaussian will lead to relatively much fill-in. On the other hand, if the arrows point downward, as for reverse Cuthill Mc Kee ordering, considerably less fill-in is obtained. Unfortunately, the reverse Cuthill Mc Kee is not a built-in method in Sepran. An option would be to first apply the built-in Cuthill Mc Kee algorithm and subsequently order the unknowns backward by means of a simple Fortran subroutine. However, although the unknowns in the Sepran meshoutputfile seem properly ordered, the plotted sparsity pattern is rather chaotic and does not match with the expected pattern. Results for the Sloan and Cuthill Mc Kee ordering algorithms are given in Chapter 7.

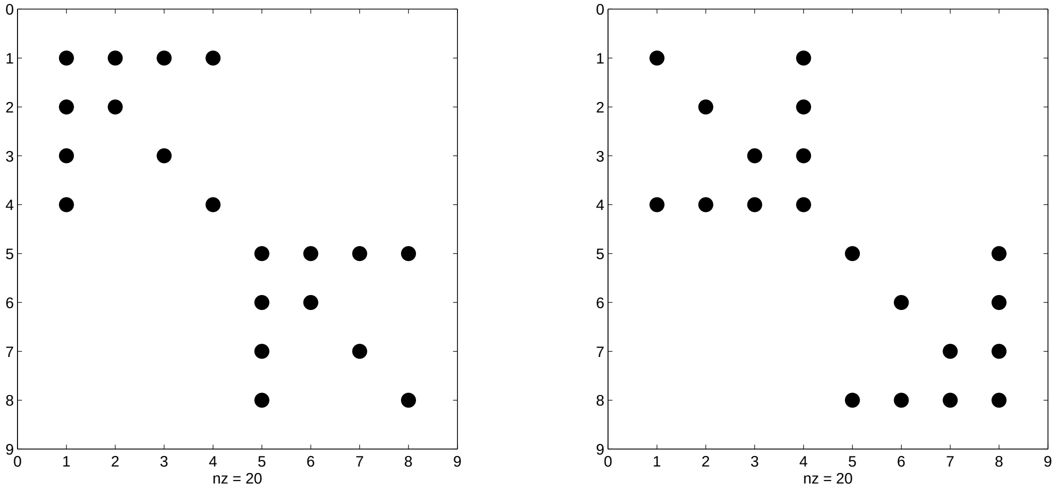


Figure 6.1: Sparsity patterns with arrows on the diagonal Left: Arrows pointing upward Right: Arrows pointing downward.

After the application of a level-set reordering method, pressure and velocity unknowns are often reordered so that the corresponding entries are separated from each other. In this case, the pressure unknowns are usually ordered last, since it is more efficient to calculate them from the knowledge of the velocity unknowns than the other way around. In a p-last ordering, all pressure unknowns appear after all velocity unknowns in the coefficient matrix. In this way, block diagonals comprising either pressure or velocity unknowns arise. In order to hold the non-zero entries in the coefficient matrix close to each other, often a p-last per level ordering is applied. This means that the pressure unknowns are ordered such that they appear per level after the velocity unknowns. Both methods have been tested and compared by M. Rehman et al [26] and G. Segal and K. Vuik [32] for the integrated method. From these tests, it can be concluded that results for p-last per level are superior to those for p-last. Results for p-last per level orderings are given in Chapter 7.

6.2.2 Level of Fill

In view of the solver problem, it may be the case that ILU preconditioning is insufficient to ensure convergence of iterative solvers. On the other hand, by allowing some additional fill-in by the Gaussian elimination process, the ILU becomes more accurate, which leads to a more efficient preconditioner. In other words, the solver performance may improve by applying Gaussian elimination to a larger pattern than the one in the original ILU [28].

The additional amount of fill-in can be regulated by introducing a so-called level of fill [28]. For each entry in the coefficient matrix, a level of fill is defined, which is updated during the Gaussian elimination process. The level of fill determines if an entry is adjusted by Gaussian elimination [28].

Rehman et al [26] state that in Sepran additional fill-in is allowed by applying ILU also to all unknowns in the neighbours of the unknowns in the original pattern. They show test results for a Stokes flow problem in a stretched backward facing step with BiCGstab, from which it can be concluded that additional fill-in for ILU preconditioning in most cases leads to an improved solver performance. Unfortunately, no literature is available in which the exact algorithm in Sepran is further specified or a motivation for the method used is given. Results for ILU preconditioning with additional fill-in are given in Chapter 7.

6.3 Preconditioners for Stokes Flow Problems

Although ILU is known as a suitable preconditioner for many applications in physics, it may not be sufficient for Stokes flow problems. Apart from some test results, such as for Sepran in [26] and [32], which are often performed on relatively coarse meshes, the literature consulted does not provide much support for ILU preconditioners for Stokes flow problems. In the literature, a wide range of different preconditioners for Stokes flow problems are suggested. One of the preconditioners for Stokes flow problems that is highly recommended in literature is the multigrid method, which is mentioned in section 6.1 [2, 10, 22, 1, 15, 21].

D. Silvester and A. Wathen [36] explain under which conditions general block preconditioners are successful for Stokes flow problems. These include modified incomplete factorisation, hierarchical basis, multigrid and domain decomposition methods. They show that if the preconditioner for the Laplacian terms is spectrally equivalent, the convergence rate of iterative solvers is independent of the mesh size. Also experimental results are plausible [36, 12]. However, they assume that the coefficient matrix is symmetric, which is not generally the case in the pressing model.

Chapter 7

Numerical Results

In Chapter 5 different stable elements for the integrated method can be found. In Chapter 6 different preconditioners and iterative solvers for discrete Stokes flow problems are presented. This chapter discusses experimental results for both TNO Glass Group's axi-symmetrical pressing[1] model and a simplified test model, for the relevant methods discussed in the previous chapters. Beforehand, it is worth studying some useful definitions:

- $\boldsymbol{r}_k = \mathcal{K}\underset{\sim}{\hat{\alpha}}_k - \boldsymbol{q}$ is the residual for the discrete Stokes flow problem in the k^{th} iteration, with $\underset{\sim}{\hat{\alpha}}_0 = \boldsymbol{0}$,

- N_k is the accumulative number of iterations up to the k^{th} time step, summed over all problems and all time steps.

- $\kappa(\mathcal{K})$ is the spectral condition number of the coefficient matrix $\mathcal{K}$ at $t = 0$,

- q is the approximate mesh refinement factor with respect to a so-called standard mesh in TNO Glass Group's pressing model, i.e. the standard mesh refined by a factor q by q.

This chapter is structured as follows. Section 7.1 numerical results of some test cases for TNO Glass Group's axi-symmetrical pressing model are examined and numerical solutions are visualised and analysed. Section 7.2 introduces a simple test model for the simulation of a pressing process time step. Then, results for some test cases for this test model are examined. The test cases are based on the theory in the previous chapters. Section 7.3 applies settings that lead to positive results for the test model to the axi-symmetrical pressing model. The result is an improved axi-symmetrical pressing model with respect to the solver performance. This chapter is concluded by examining some numerical solutions for the improved axi-symmetric pressing model and comparing them to results in section 7.1.

[1] That is, TNO Glass Group's pressing model in which the option "axi-symmetric" is switched on (see section 3.5)

7.1 Results for TNO Glass Group's Axi-Symmetrical Pressing Model

7.1.1 Solver Performance for the Stokes Flow Problem

In section 1.3 an example of the solver performance for BiCGstab with ILU preconditioning is given. From this example it can be clearly seen that application of mesh refinement for the discrete Stokes flow problems produces an excessive increase in the number of iterations. Chapter 5 analyses the discretisation methods applied in TNO Glass Group's pressing model, which leads to the conclusion that they are stable and suitable for their stated purpose. The objective of this section is to analyse the sensitivity of the solver performance in TNO Glass Group's pressing model to changes in numerical or modelling settings.

For the representation of the results, the number of iterations for the initial Stokes flow problem, N_1, and the accumulative number of iterations up to the 90th time step, N_{90}, are considered. Since no time derivatives occur in the Stokes flow problems, a fixed time step $\Delta t = \frac{1}{900}$ is chosen. Further on in this section, the Stokes flow problems are decoupled from the energy exchange problem and glass-air interface problem, so that neither the material parameters in the Stokes flow equations are time dependent. In this case, the size of the time step will not influence the solver performance, which only slightly changes in time, as in each time step a different flow domain and mesh are considered. Furthermore, it should be mentioned that the 90th time step corresponds to a point in time ($t = 0.1$) at which the glass melt has not reached the bend in the mould-plunger construction yet.

Condition Number

In section 1.3 it is mentioned that the condition number of the coefficient matrix in initial discrete Stokes flow problems in TNO Glass Group's pressing model is substantial. This condition number is determined from the diagonal of the LU factorisation (min/max). In section 3.5 it is explained that the condition number significantly increases because of a large numerical value of the wall friction coefficient. Experiments confirm that the condition number is also sensitive to the numerical value of the viscosity. A larger condition number means that the coefficient matrix is closer to singular, hence larger numerical errors will occur and iterative solvers that are not robust enough will fail to converge.

Table 7.1 sets out the condition numbers of the coefficient matrices in initial discrete Stokes flow problems for BiCGstab with ILU preconditioning and different mesh densities. As verified in section 6.1, the condition number is proportional to $O(h^{-2})$, and hence to $O(q^2)$. Although the increase in the condition number is always quadratic, the increase in the number of iterations is linear for coarse meshes, but much larger for fine meshes. Moreover, the number of iterations for $q = \frac{1}{2}$ is rather small, whereas the condition number for this mesh density is already considerable. Further on it can be observed that the solver problem may as well occur to the same extent for much smaller condition numbers, from which it may be concluded that the solver behaviour is not directly related to the condition number.

Table 7.1: Numerical data for BiCGstab with ILU preconditioning applied to the discrete Stokes flow problem in TNO Glass Group's axi-symmetrical pressing model.

q	$\|\|\boldsymbol{r}_0\|\|_2$	N_1	N_{90}	$\kappa_{t=0}$
$\frac{1}{2}$	2.73E-1	12	1362	3.70E22
1	3.83E-1	27	2760	1.55E23
2	5.39E-1	164	13647	6.37E23
3	6.59E-1	1752	109930	1.45E24

Iterative Solvers

The solver problem also occurs for different iterative solvers from BiCGstab. For example for GMRES with ILU preconditioning on the standard mesh, 74 iterations until convergence are observed for the initial Stokes flow problem and 6411 accumulated iterations at the 90th time step. If the standard mesh is one time two by two refined (q = 2), even 3618 iterations until convergence are observed for the initial Stokes flow problem and after a few time steps, GMRES reaches the maximum number of iterations allowed. The modelling settings are the same as for BiCGstab in section 1.3. In conclusion, the solver problem is even worse for GMRES than for BiCGstab.

Stop Criterium

From Table 7.1, it can be concluded that the normalised initial residual $||\boldsymbol{r}_0||$ increases with the mesh density. Apparently, the residual depends on the mesh size. Since on the other hand the tolerance remains independent of the mesh size, an absolute stop criterium may be too strict for fine meshes. If the solver cannot reduce the residual anymore on fine meshes to below a small fixed tolerance, it will finally terminate, whereas the smallest residual obtained may be quite reasonable in view of the small mesh size. This is an inducement to employ a relative stop criterium with respect to the initial residual. However, from Table 7.1 it can be seen that the stop criterium hardly makes any difference in the relative increase in N_k. Note that the relative stop criterium is slightly stricter, as the initial residuals are smaller than one. Thus, the stop criterium used does not contribute to the solver problem.

Simplification of Stokes Flow Problems

Discrete Stokes flow problems in themselves for the discretisation methods concerned are not expected to give rise to problems of such nature as encountered in the previous sections. However, the flow of glass in a pressing process is essentially influenced by three characteristic physical phenomena involved.

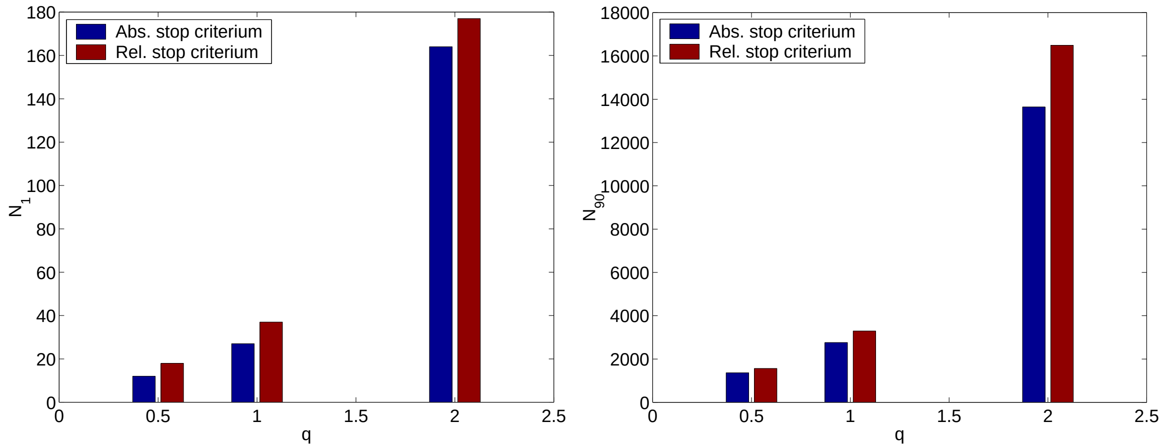


Figure 7.1: Accumulative number of iterations for the BiCGstab solver with ILU preconditioning in the pressing model. The blue bars represent the results for an absolute stop criterium and the red bars represent the results for a relative stop criterium with respect to the initial residual. The tolerance is ϵ = 1E-3.

1. The flow problem is temperature dependent, so that it is influenced by the energy exchange during the pressing process.
2. Glass and air are separated by a sharp interface.
3. The flow domain is changing as a result of the plunger movement.

Phenomenon 1 is not expected to give any problems, since at each time step a temperature distribution is calculated and the corresponding numerical values are substituted into the discrete Stokes flow problem. In this way, each time step a Stokes flow problem with a slightly different viscosity is solved. However, although after substitution the temperature will not explicitly appear in the flow problem anymore, the viscosity will still be dependent on the position. The modelling of phenomenon 2 involves discontinuities in material properties and BCs. This may lead to extreme gradients or singular behaviour in the numerical model, which can affect the condition of the discrete Stokes flow problem considerably. Such behaviour can particularly be expected at the glass-air interface near the equipment boundary. This possibility is further analysed in Appendix B. Finally, from the previous results it can be concluded that phenomenon 3 does not influence the solver problem, since the number of iterations in the first time step, N_1, is almost the same as the average number of iterations up to the 90th time step, $\frac{N_{90}}{90}$.

The influence of phenomena 1 and 2 can be verified by excluding the phenomena in the pressing model. Phenomenon 1 can easily be omitted by taking a constant viscosity. If also the BCs are the same on each side of the glass-air interface, the flow transition will be smooth at the interface. For example, let the viscosity of the glass melt be equal to the viscosity of the fictive fluid and assume free slip conditions for both glass and fictive fluid. A test case for this example with BiCGstab and ILU preconditioning on the standard mesh ($q = 1$) leads to 32 iterations until convergence for the initial

Stokes flow problem and 3195 accumulated iterations at the 90th time step. For $q = 2$, $N_1 = 141$ and $N_{90} = 13491$, respectively. So, in comparison to the original problem (see for example Table 7.1), hardly any improvement in the solver performance is observed. A full simplification of a Stokes flow problem in a pressing process time step is considered in section 7.2.

Implementation of Boundary Conditions

In section 3.5 it is mentioned that mixed BCs are applied on all boundaries in TNO's pressing model. This leads to the choice of a large numerical value for the friction coefficient in equation (3.5.1), and hence to a significant increase in the condition number.

Two obvious possibilities are available to test the influence of the large friction coefficient on the solver performance. Firstly, the solver performance can be tested for a different pressing problem, for which mixed BCs are imposed instead of Dirichlet BCs. This can easily be done by decreasing the friction coefficient in the original problem. Secondly, Dirichlet BCs can explicitly be prescribed after each time step instead of an approximation by means of an extremely large friction coefficient. This can be achieved by means of the procedure explained in section 4.10. In this way, the same problems are solved, but the method is computationally relatively inefficient, since after each time step a new projection matrix has to be constructed.

Results for the second method are shown in Table 7.2 and Figure 7.2. The solver problem for Dirichlet BCs seems to be even worse than for mixed BCs with a large friction coefficient, as for Dirichlet BCs the increase in iterations from $q = 2$ to $q = 3$ is approximately two times as large as for mixed BCs. From the data for Dirichlet BCs in Table 7.2, it can be seen that for $q = 3$ almost 20 times more iterations are required than for $q = 2$. Table 7.2 also reveals that the solver performance not necessarily improves if the condition number is reduced; in contrast to the still perceptible, excessive increase in the number of iterations, the condition number is shrunk by a factor 10^6.

Table 7.2: Numerical data for BiCGstab with ILU preconditioning applied to the discrete Stokes flow problem in TNO Glass Group's axi-symmetrical pressing model for an implementation of Dirichlet BCs instead of mixed BCs

q	$\|\|\boldsymbol{r}_0\|\|_2$	$N_{t=0}$	$N_{t=0.1}$	$\kappa_{t=0}$
$\frac{1}{2}$	4.24E-1	10	1150	1.513E16
1	4.24E-1	18	2000	1.321E17
2	5.97E-1	131	10901	8.547E17
3	7.30E-1	3760	191177	3.984E18

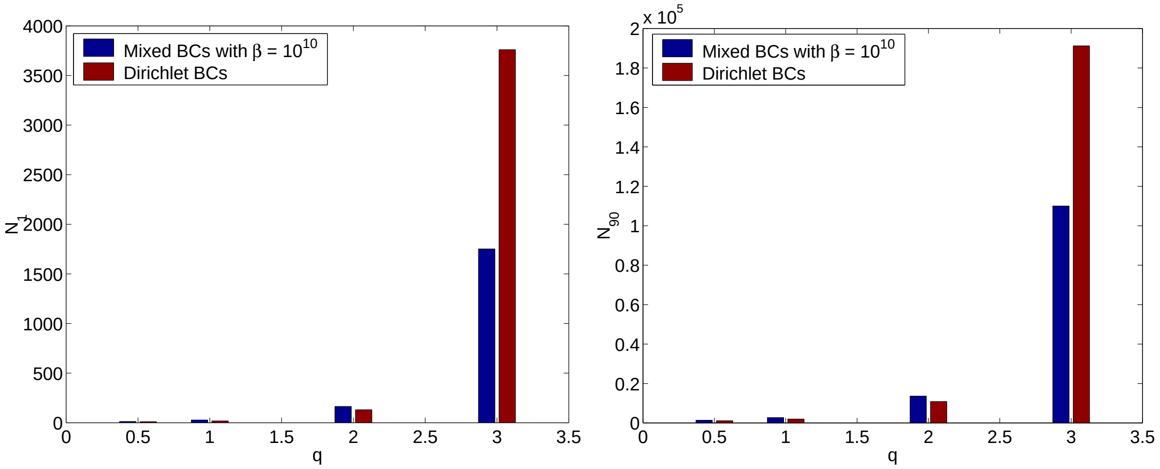


Figure 7.2: Accumulative number of iterations for the BiCGstab solver with ILU preconditioning in the pressing model. The blue bars represent the results for Mixed BCs with $\beta = 10^{10}$ and the red bars represent the results for Dirichlet BCs. An absolute stop criterium with tolerance ϵ = 1E-3 is used.

Mesh Distribution

From Figure 5.3a- 5.3d in section 5.7 it can be seen that the elements are not uniformly distributed over the width of the flow domain. In order to calculate high gradients at the equipment walls, a noticeable increased mesh density prevails at these locations. This may lead to unexpected numerical results and in the worst case to the violation of inf-sup condition (5.3.5). The influence of this mesh distribution on the solver performance can easily be observed by solving the pressing model for a uniform mesh distribution over the width of the flow domain. The results are given in Figure 7.3. Obviously, the solver performance depends on the mesh distribution, but the increase in iterations for mesh refinement is much more severe for the uniform mesh distribution than for the original mesh distribution.

Naturally, this observation raises the question, for which mesh distribution the number of iterations will be minimal. To this purpose, introduce a so-called mesh distribution factor σ for the pressing model, which represents the ratio of the number of elements at the middle of the mould plunger construction to the number of elements at the equipment wall, measured over the width of the flow domain. For the original mesh $\sigma = 0.2$. In Figure 7.4 the number of iterations for the BiCGstab solver with ILU preconditioning on the standard mesh is given for different mesh distributions. It appears that on the original standard mesh the number of iterations is quite minimal. It also becomes clear that N_{90} is roughly of the same order of magnitude for all mesh distributions. Therefore, it cannot be expected that the solver problem can be significantly reduced by applying a different mesh density.

Remarkably, N_k also seems to be more stable in time for larger mesh distribution factors, since N_{90} becomes closer to 90 times N_1 as $\sigma \to 1$. This property can still be observed for $\sigma = 0.2$, but for $\sigma = 0.1$, N_{90} is only approximately 10 times N_1. This stability in time is an important property, since

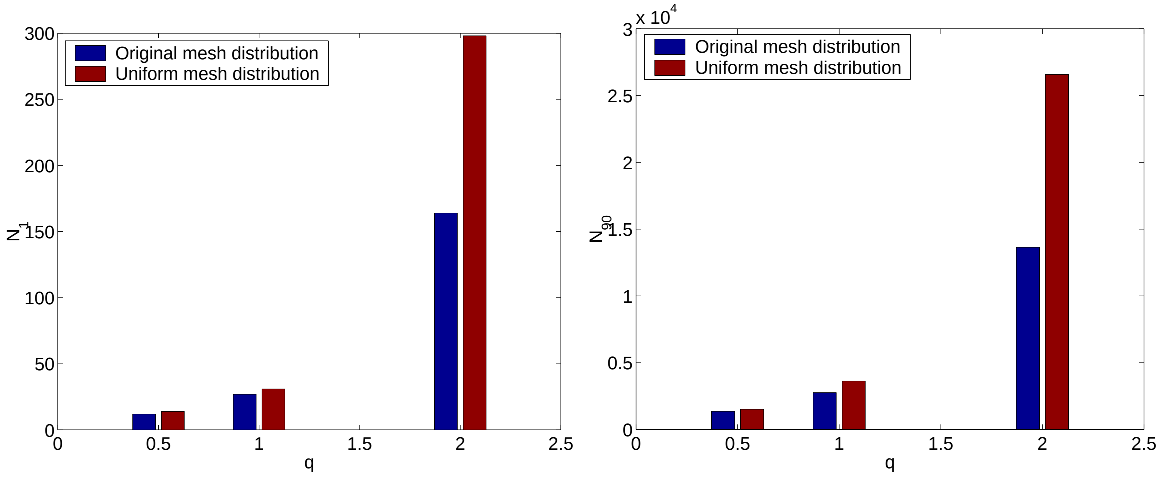


Figure 7.3: Accumulative number of iterations for the BiCGstab solver with ILU preconditioning in the pressing model. The blue bars represent the results for the original mesh distribution and the red bars represent the results for a uniform mesh distribution in normal direction. An absolute stop criterium with tolerance ϵ = 1E-3 is used.

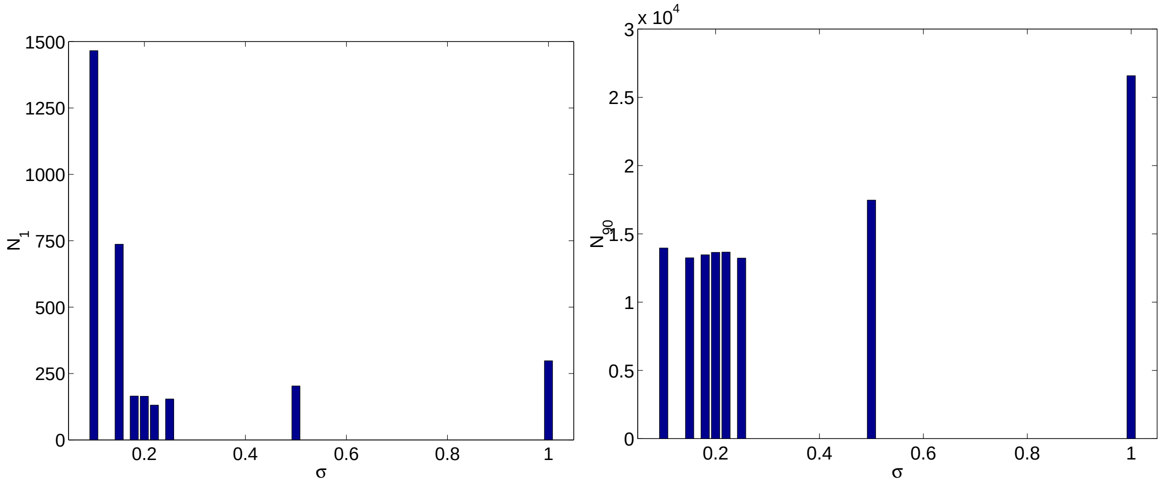


Figure 7.4: Accumulative number of iterations for the BiCGstab solver with ILU preconditioning in the pressing model with q = 1 for different mesh distributions. An absolute stop criterium with tolerance ϵ = 1E-3 is used.

it means that the solver problem does not increase as the elements are flattened or stretched during the mesh deformation. Although this property is still preserved for σ = 0.2 up to 90 time steps, clear changes in the solver performance can become perceptible after for example 900 time steps. Yet these changes are in most cases not alarming and the precise effect of mesh deformation on these results is not clear, since also the problem itself is changing because of the decelerating plunger, moving interface and increasing viscosity.

The influence of mesh stretching on the solver performance has been tested in Sepran by Rehman et al [26]. They reveal that if the mesh is stretched too much, this could result in a enormous increase in iterations or even lead to termination of the solver.

The phenomenon of mesh stretching is also mentioned in section 3.5. Although it is not exactly clear to which extent the solver performance is influenced by mesh deformation, it is certain that mesh deformation is not the main cause of the solver problem. In fact, it is not expected that element deformation during the pressing process has an abominable effect on the solver performance. Moreover, existing mesh adaption algorithms that avoid mesh deformation due to a changing domain are known for their relative inefficiency (see for example the Sepran user manual [31]). Therefore, the mesh adaption algorithm in TNO Glass Group's pressing model is not considered for improvement. In section 7.2 the solver performance is tested on a fixed mesh that partitions the flow domain into square cells, so that mesh deformation does not play any role.

7.1.2 Numerical Solutions

In this section numerical solutions of the pressing model are plotted to observe how they behave, in particular in view of mesh refinement. All problems are solved by the BiCGstab solver with ILU preconditioning. An absolute stop criterium with $\epsilon = 1\text{E-}3$ is used.

In Figure 7.5a-7.5d, Figure 7.6a-7.6d and Figure 7.7a-7.7d, pictures of the flow of glass and air for different mesh densities at different times are given. From these pictures, it appears that the glass-air transition becomes thinner and smoother after mesh refinement. The location of the glass-air interface at each time step only slightly changes if the mesh is refined from $q = \frac{1}{2}$ to $q = 1$. Note that the glass-air interface stretches out somewhat further near the mould than near the plunger. Since there is no gravity, this is apparently the result of the plunger pressing the glass away. It can also be observed that for $q = 1$ some air is left at $t = 1.5$, whereas for coarser meshes, all air in the flow domain has disappeared by this time. In conclusion, numerical solutions improve perceptibly if mesh refinement is applied, which confirms the necessity of improving the solver performance for fine meshes. The figures do not indicate any suspicious behaviour in the flow profile after mesh refinement that could reasonably lead to an excessive increase in the number of iterations.

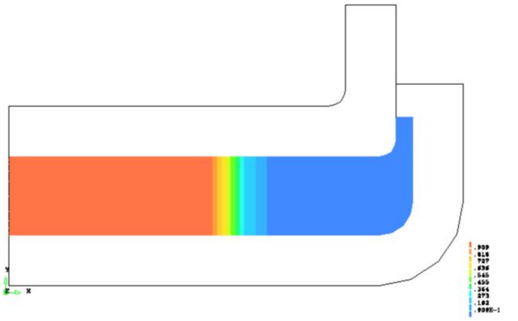


Figure 7.5a: Flow of glass and air in TNO Glass Group's axi-symmetrical pressing model for $q = \frac{1}{2}$ at $t = 0$ s. Glass is orange and air is blue.

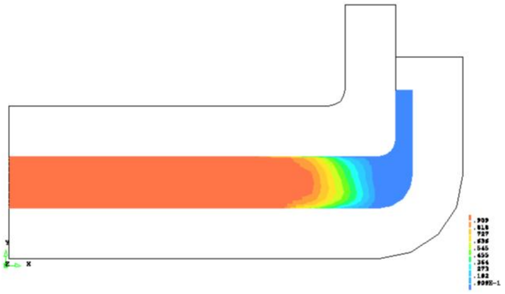


Figure 7.5b: Flow of glass and air in TNO Glass Group's axi-symmetrical pressing model for $q = \frac{1}{2}$ at $t = 0.5$ s. Glass is orange and air is blue.

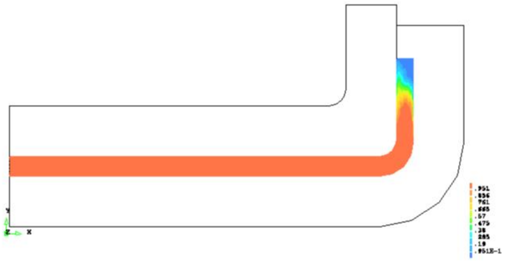


Figure 7.5c: Flow of glass and air in TNO Glass Group's axi-symmetrical pressing model for $q = \frac{1}{2}$ at $t = 1.0$ s. Glass is orange and air is blue.

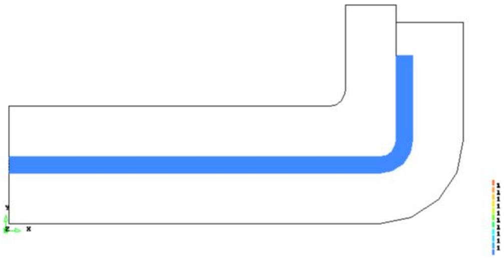


Figure 7.5d: Flow of glass and air in TNO Glass Group's axi-symmetrical pressing model for $q = \frac{1}{2}$ at $t = 1.5$ s. Only glass is left.

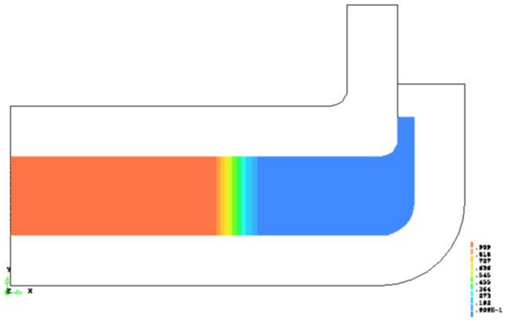


Figure 7.6a: Flow of glass and air in TNO Glass Group’s axi-symmetrical pressing model for $q = 1$ at $t = 0$ s. Glass is orange and air is blue.

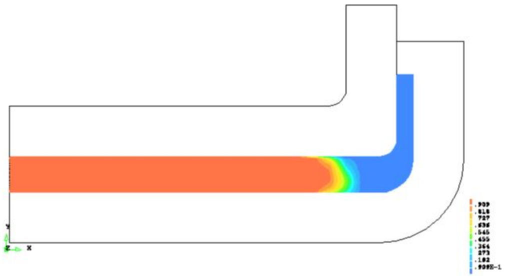


Figure 7.6b: Flow of glass and air in TNO Glass Group’s axi-symmetrical pressing model for $q = 1$ at $t = 0.5$ s. Glass is orange and air is blue.

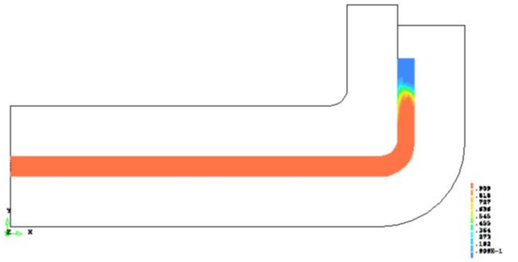


Figure 7.6c: Flow of glass and air in TNO Glass Group’s axi-symmetrical pressing model for $q = 1$ at $t = 1.0$ s. Glass is orange and air is blue.

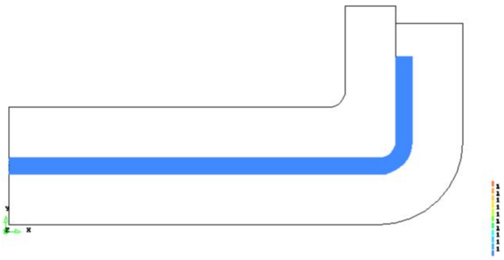


Figure 7.6d: Flow of glass and air in TNO Glass Group’s axi-symmetrical pressing model for $q = 1$ at $t = 1.5$ s. Only glass is left.

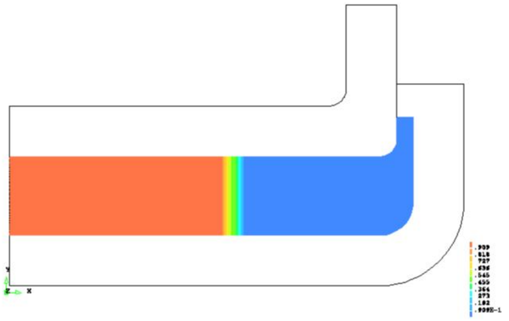


Figure 7.7a: Flow of glass and air in TNO Glass Group's axi-symmetrical pressing model for q = 2 at t = 0 s. Glass is orange and air is blue.

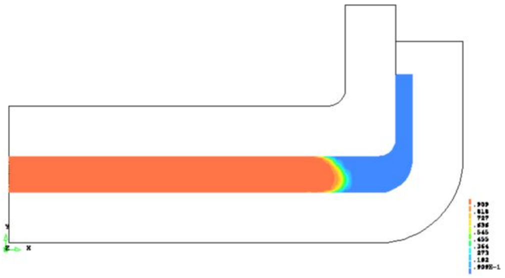


Figure 7.7b: Flow of glass and air in TNO Glass Group's axi-symmetrical pressing model for q = 2 at t = 0.5 s. Glass is orange and air is blue.

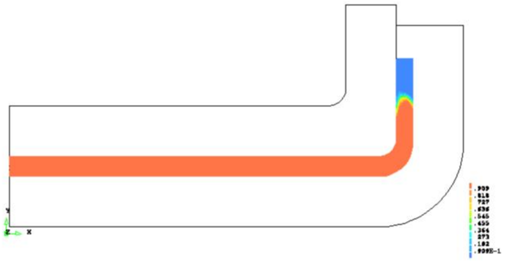


Figure 7.7c: Flow of glass and air in TNO Glass Group's axi-symmetrical pressing model for q = 2 at t = 1.0 s. Glass is orange and air is blue.

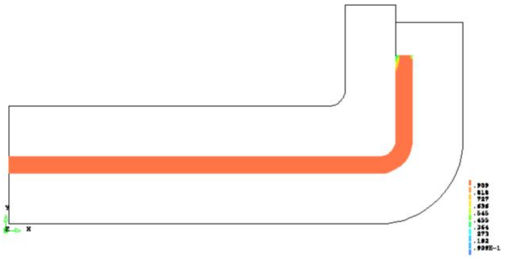


Figure 7.7d: Flow of glass and air in TNO Glass Group's axi-symmetrical pressing model for q = 2 at t = 1.5 s. Only glass is left.

In Figure 7.8a-7.8d pictures of the temperature distribution in the glass pressing construction for $q = 1$ at different times are given. The temperature ranges between 550 °C and 1000 °C. The glass temperature slowly expands into the equipment domain, but only a thin boundary layer becomes hotter than 550 °C within 1.5 s. On the other hand, the glass rapidly heats the air and by $t = 1.0$ s, the entire cavity in the glass pressing construction takes the glass temperature.

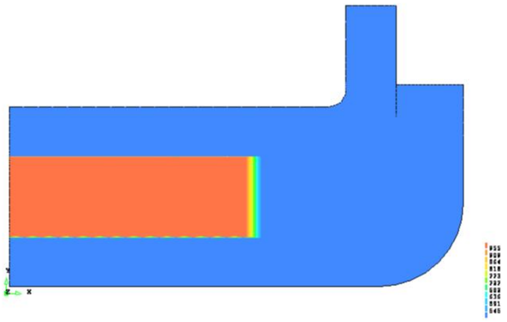


Figure 7.8a: Temperature distribution in TNO Glass Group's axi-symmetrical pressing model for $q = 1$ at $t = 0$ s. The temperature ranges from ca 550 °C (blue) to ca 950 °C (orange).

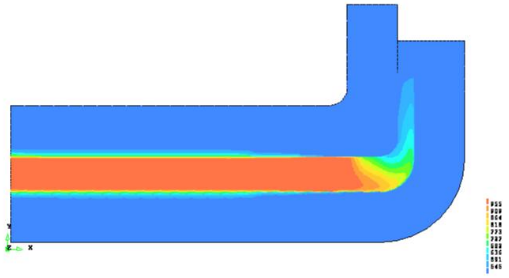


Figure 7.8b: Temperature distribution in TNO Glass Group's axi-symmetrical pressing model for $q = 1$ at $t = 0.5$ s. The temperature ranges from ca 550 °C (blue) to ca 950 °C (orange).

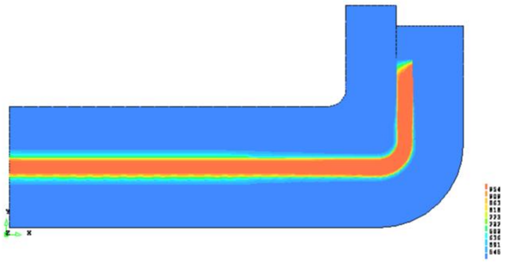


Figure 7.8c: Temperature distribution in TNO Glass Group's axi-symmetrical pressing model for $q = 1$ at $t = 1.0$ s. The temperature ranges from ca 550 °C (blue) to ca 950 °C (orange).

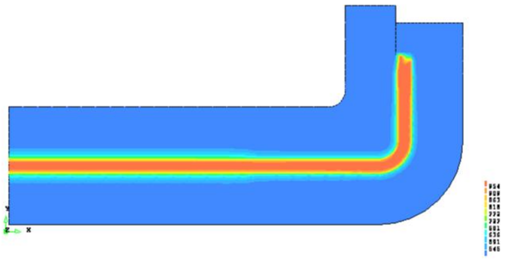


Figure 7.8d: Temperature distribution in TNO Glass Group's axi-symmetrical pressing model for $q = 1$ at $t = 1.5$ s. The temperature ranges from ca 550 °C (blue) to ca 950 °C (orange).

In order to verify the accuracy of numerical solutions, mass conservation of glass is examined. Since incompressibility is assumed, also the glass volume should be conserved. In Figure 7.9 the glass volume is plotted as a function of time for different mesh densities. As expected, numerical solutions are more accurate and correct themselves more quickly for finer meshes. For comparison: for $q = \frac{1}{2}$ the volume difference is more than 1 percent on average, whereas for $q = 2$ the volume difference is ca 0.12 percent on average. Furthermore, for $q = \frac{1}{2}$ the volume gain is even ca 3 percent at $t = 1$. This expresses itself for instance in Figure 7.5c, where the glass flow front has travelled further than the glass flow front in Figure 7.6c and Figure 7.7c.

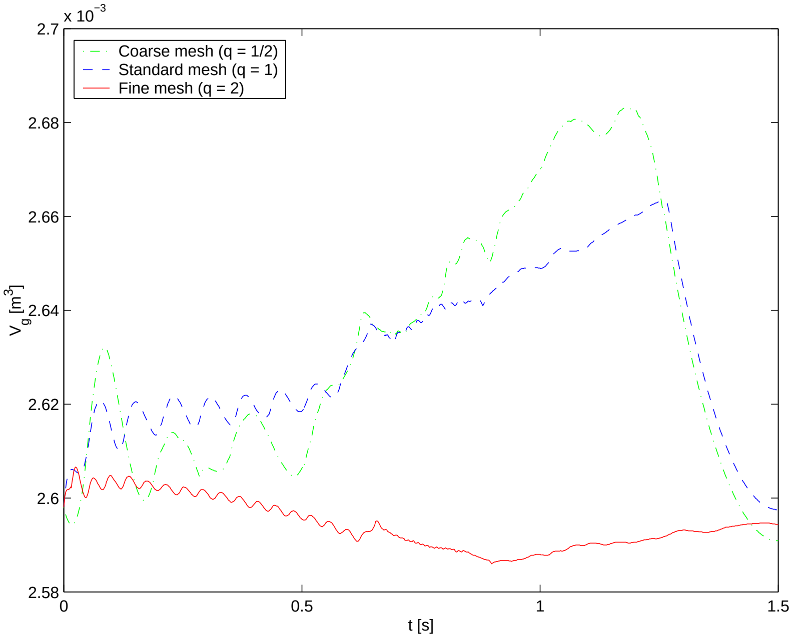


Figure 7.9: Glass volume (V_g [m^3]) conservation

From the results in this section, it does not appear that any circumstances in the pressing model arise that may lead to an extensive increase in iterations as the mesh is refined. The next step is to test the solver performance for a most simplified pressing model. This model is again implemented in Sepran, using a minimum amount of source code.

7.2 Results for a Test Model for the Simulation of a Pressing Process Time Step

In this section the solver performance is analysed for a simple pressing model (see Figure 7.10). In this test model a pressing process time step for a rectangular domain is considered. The domain contains only one flow medium. On top of the domain there is a constant inflow, which resembles a plunger pressing the medium from above at a fixed time. The medium within the rectangle can only flow out through the right side, where there is free flow. Furthermore, on top and on the bottom, no slip conditions are prescribed, whereas the left side represents a symmetry-axis. Both the axi-symmetric problem, as well as the Cartesian version are considered. The finite element mesh is constructed as in Figure 7.11.

In this section, the following numerical settings are used, unless stated otherwise. The flow domain is squared and a uniform mesh is constructed by partitioning the domain into square cells of size $h \times h$ for some h, and subsequently subdividing the cells into two triangular Mini-elements (see Figure 7.11). The bubble functions in the Mini-elements are eliminated by static condensation at element level as explained in section 5.6. The BiCGstab solver with ILU preconditioning is applied. Numerical solutions are accepted if an absolute stop criterium with ϵ = 1E-10 is satisfied.

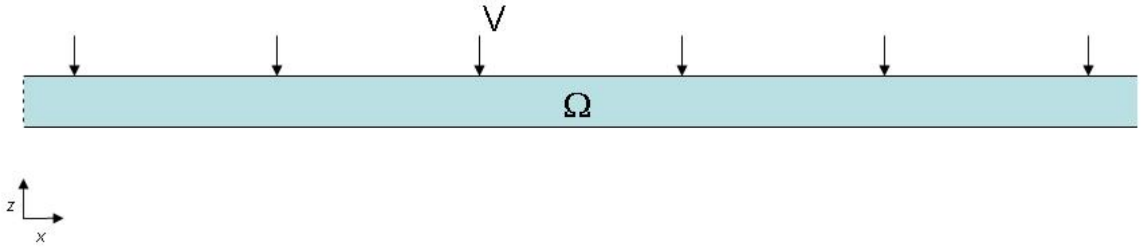


Figure 7.10: Test problem for the simulation of a pressing process time step

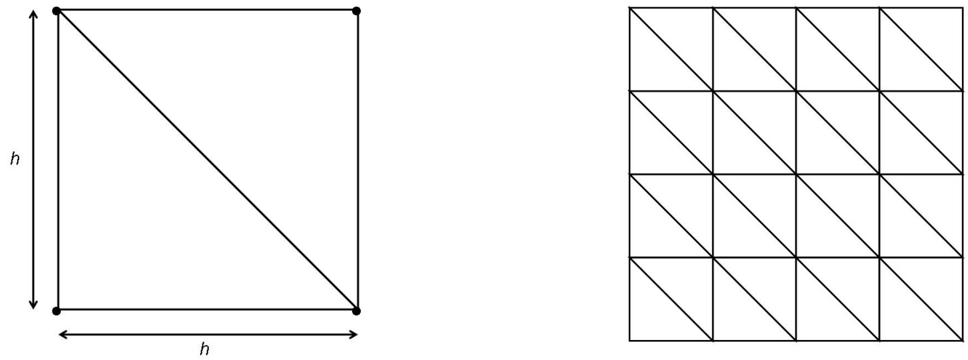


Figure 7.11: Uniform partitioning of a rectangular domain into right-angled triangular Mini-elements. Left: Subdivision of a $h \times h$ square cell into two right-angled triangular Mini-elements. Right: Mesh consisting of 32 Mini-elements.

7.2.1 Simulation of an Axi-Symmetric Glass Pressing Process Time Step

First of all, consider an axi-symmetric glass pressing time step. Let the material parameters be equal to their typical values in section 3.2, i.e. $\rho_0 = 2.5 \cdot 10^3$ kg m^{-3}, $\mu_0 = 10^4$ kg m^{-1}s^{-1}. In addition, let the height of the domain be equal to the characteristic length $L = 0.01$ m. For the plunger velocity, the value $V = -1.26 \cdot 10^{-1}$ m s^{-1} is chosen. So, the Reynolds number is Re $= 3.15 \cdot 10^{-4}$.

The flow problem is solved for both the dimensional and the dimensionless case on a mesh consisting of 45056 right-angled triangular Mini-elements (a partition into 32 by 704 square cells, as the length of the domain is 0.22 m). Figure 7.12a-7.12c shows the results for the dimensional problem and Figure 7.12a-7.12c shows the results for the dimensionless problem. Apart from the dimensions, both the flow and pressure profiles are rather similar. In Table 7.3 numerical data for both problems is set out. For both problems approximately the same amounts of iterations are required until convergence. Overall these numbers are slightly higher for the dimensionless model, but apparently this is the result of the larger initial residuals. Furthermore, it can be observed that the condition number for the dimensionless model are roughly a factor 10^5 smaller than the condition number for the dimensional model. However, this cannot be seen back in the solver performance, since in comparison the iterations increases by the same amounts and the solver terminates for the same mesh densities. That the increase in iterations is actually excessive can be concluded by comparing the CPU time for BiCGstab to the CPU time for the direct method. On all meshes, the iterative solver takes longer than the direct method, especially on fine meshes, whereas iterative solvers have been developed because of the relative inefficiency of direct methods for fine meshes. For 180224 elements, the iterative solver even takes ca 40 times as much CPU time to solve the problems as the direct method. Note that for 720896 elements, the direct method executes because of a shortage on storage capacity. For the same reason, the condition number could not be computed.

Table 7.3: Numerical data for the test model for the simulation of a glass pressing process time step

elements		2816	11264	45056	180224	720896
dimensional model	iterations	20	36	153	7985	-
	initial residual	1.14	1.60	2.26	3.20	4.53
	condition number	3.94E19	3.15E20	2.53E21	2.02E22	-
	CPU time	1.68E-1	7.73E-1	8.27	1.39E3	-
	CPU time for direct method	1.37E-1	5.80E-1	3.61	34.6	-
dimensionless model	iterations	22	36	168	9133	-
	initial residual	9.01	12.7	18.0	25.4	35.9
	condition number	3.97E14	3.18E15	2.55E16	2.04E17	-
	CPU time	2.08E-1	1.05	13.4	1.74E3	-
	CPU time for direct method	1.45E-1	6.27E-1	3.98	37.6	-

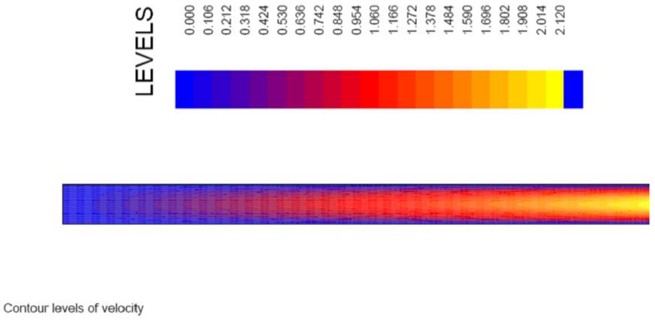


Figure 7.12a: Flow velocity solution in *x*-direction for the test model for the simulation of a glass pressing process time step.

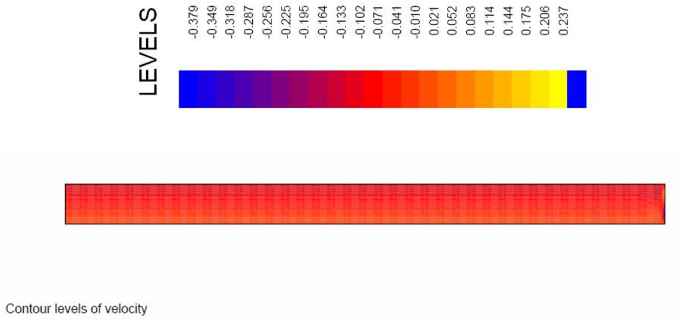


Figure 7.12b: Flow velocity solution in *z*-direction for the test model for the simulation of a glass pressing process time step.

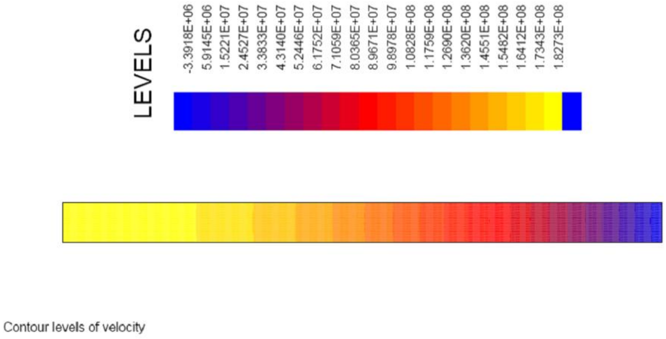


Figure 7.12c: Pressure solution for the test model for the simulation of a glass pressing process time step.

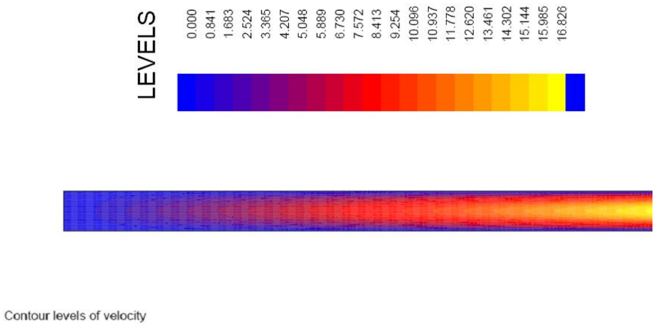


Figure 7.13a: Flow velocity solution in *x*-direction for the test model for the simulation of a dimensionless glass pressing process time step.

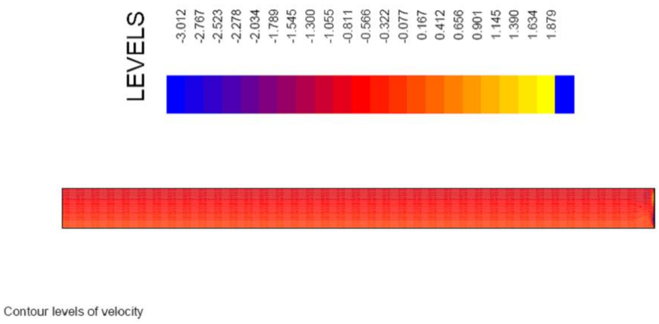


Figure 7.13b: Flow velocity solution in *z*-direction for the test model for the simulation of a dimensionless glass pressing process time step.

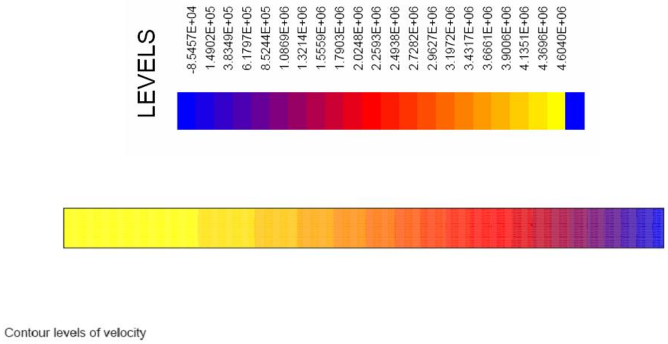


Figure 7.13c: Pressure solution for the test model for the simulation of a dimensionless glass pressing process time step.

7.2.2 Different Preconditioners

For the remainder of this section, consider a dimensionless pressing process on a two-dimensional, Cartesian, squared domain. Let $\mu = \mathrm{Re}^{-1} = 2 \cdot 10^{-2}$. This value has been experimentally determined to give a small condition number for a large range of test cases.

The following preconditioners are tested: no preconditioner, Gauss-Seidel, Eisenstat and ILU (see the Sepran manual [31]). Figure 7.14 gives the corresponding numbers of iterations for 512 elements (the condition number is only 113 in this case). Obviously, this number is smallest for ILU preconditioning. If the mesh is a factor 2 by 2 refined, BiCGstab diverges for Gauss-Seidel and Eisenstat preconditioning, whereas ILU results in 41 iterations and without preconditioning not less than 1781 iterations are required. The corresponding CPU times for the refined mesh are 1.65 without preconditioning and 0.20 for ILU. In conclusion, the ILU preconditioner is the best option among the aforementioned ones.

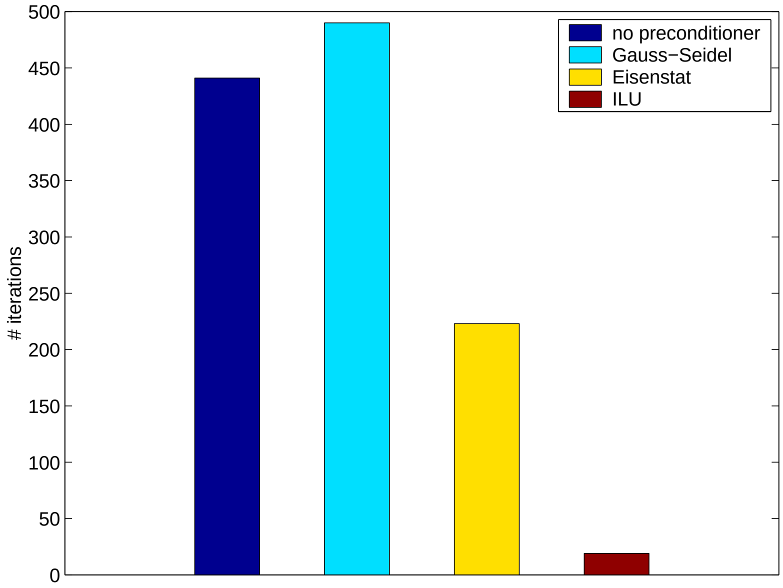


Figure 7.14: Numbers of iterations for 512 elements and different preconditioners.

7.2.3 Different Orderings of the Unknowns

The solver performance is tested for different orderings of the unknowns. Different ordering methods are discussed in section 6.2.1. Let N be the number of iterations until convergence and n_e the number of elements in the mesh. Figure 7.15 shows the increase in iterations as the mesh is refined for different ordering methods. For all meshes Cuthill Mc Kee gives slightly better results than Sloan. In addition, in all cases p-last/level leads to a better solver performance for both ordering methods. Table 7.4 also gives the CPU time. Indeed, the CPU time is smallest for Cuthill Mc Kee with p-last/level. Note that the decrease in CPU time for the Reverse Cuthill Mc Kee algorithm if p-last/level is applied is fairly insignificant. In conclusion, the number of iterations can be reduced by using Cuthill Mc Kee instead of Sloan. However, although some improvement is gained, the iterative solver still terminates as the mesh is refined from 32768 to 131072.

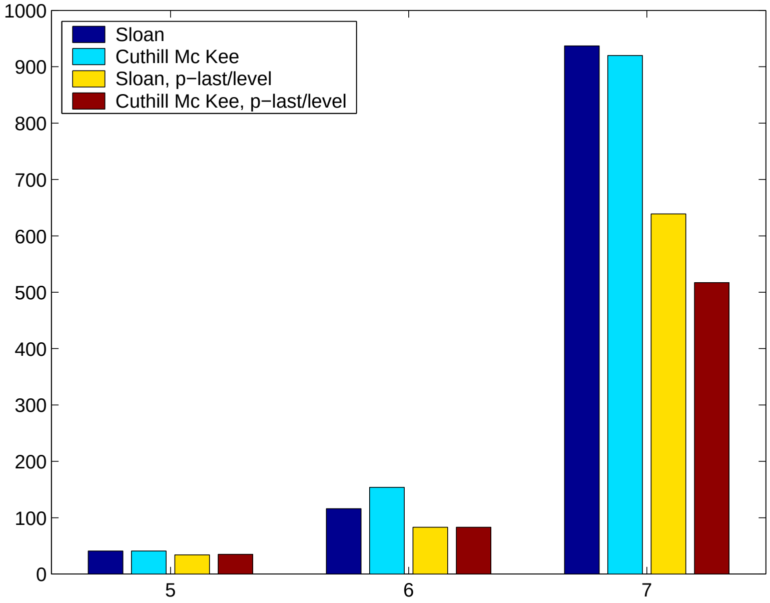


Figure 7.15: Numbers of iterations N for different orderings of the unknowns and different values of $\log_4(n_e/2)$.

Table 7.4: Number of iterations and CPU time for different orderings of the unknowns and different mesh densities.

		n_e	2048	8192	32768	131072
Sloan		N	41	116	937	-
		CPU	2.04E-1	1.77	50.9	-
Sloan	p-last/level	N	34	83	639	-
		CPU	1.44E-1	1.03	21.3	-
Cuthill Mc Kee		N	41	154	920	-
		CPU	1.56E-1	1.44	30.5	-
Cuthill Mc Kee	p-last/level	N	35	83	517	-
		CPU	1.40E-1	8.65E-1	17.3	-

Figure 7.16a-7.16c show the sparsity pattern of the coefficient matrix for different ordering methods. The domain is partitioned into 512 elements, which corresponds to 784 unknowns. Apparently, in the coefficient matrix without reordering, the pressure unknowns come before the velocity unknowns.

It can be seen that the differences between the sparsity patterns for Sloan and Cuthill Mc Kee orderings are rather small. The in section 6.2.1 suggested “arrows” on the diagonal in the sparsity pattern corresponding to Cuthill Mc Kee ordering do not seem to appear in Figure 7.16c, and appear only to a small extent in Figure 7.17b. Figure 7.16b-7.16c show that the non-zero entries are clustered closely together. On the other hand, Figure 7.17a-7.17b illustrate that if additionally p-last/level ordering is applied, the clusters are spread over a larger number of smaller clusters. This is the result of the pressure unknowns being separated from the velocity unknowns.

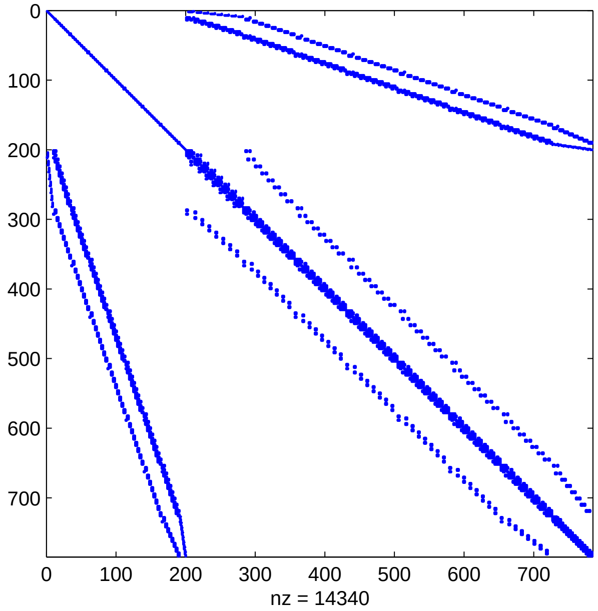
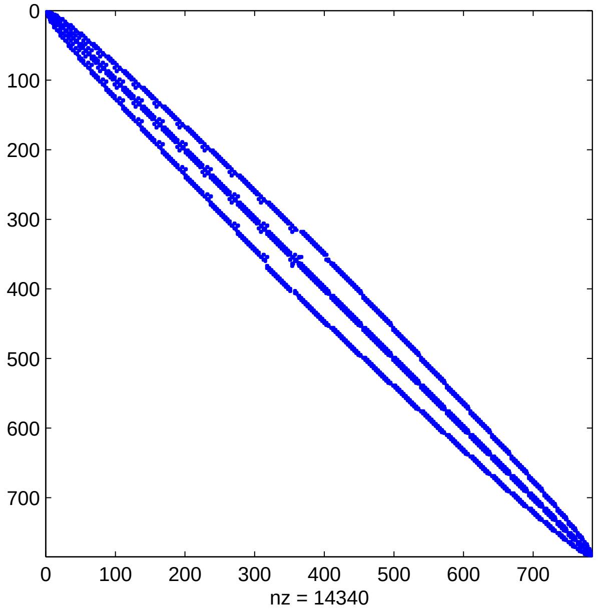


Figure 7.16a: Sparsity pattern of coefficient matrix without reordering for 512 elements.



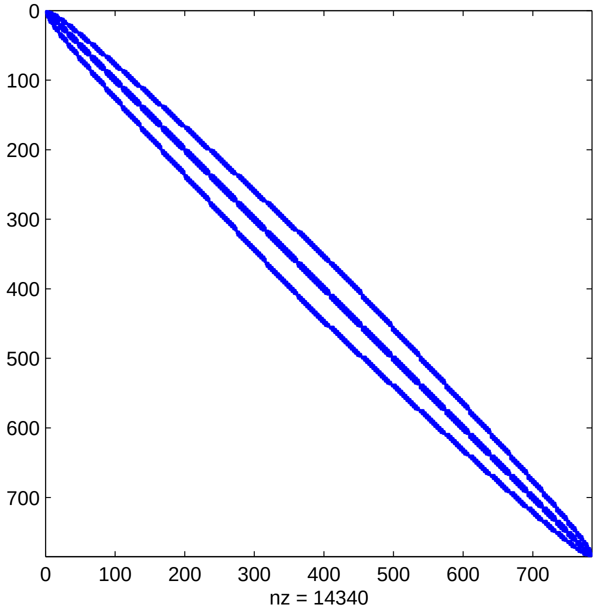


Figure 7.16b: Sparsity pattern of coefficient matrix with Sloan reordering for 512 elements.

Figure 7.16c: Sparsity pattern of coefficient matrix with Cuthill Mc Kee reordering for 512 elements

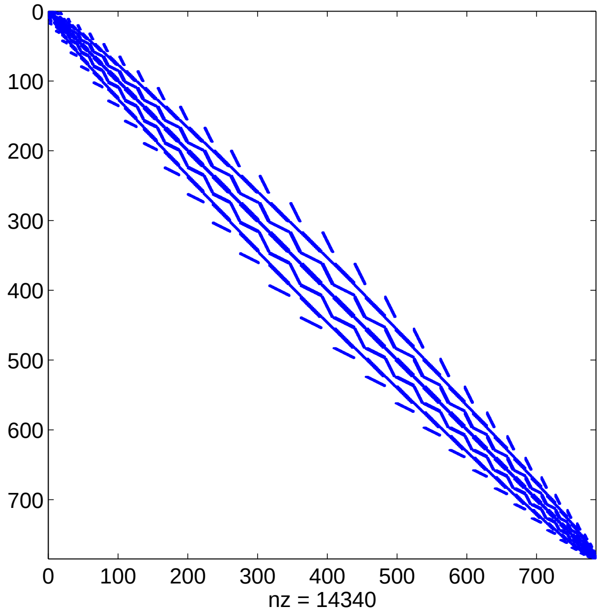


Figure 7.17a: Sparsity pattern of coefficient matrix with Sloan and p-last/level reordering for 512 elements.

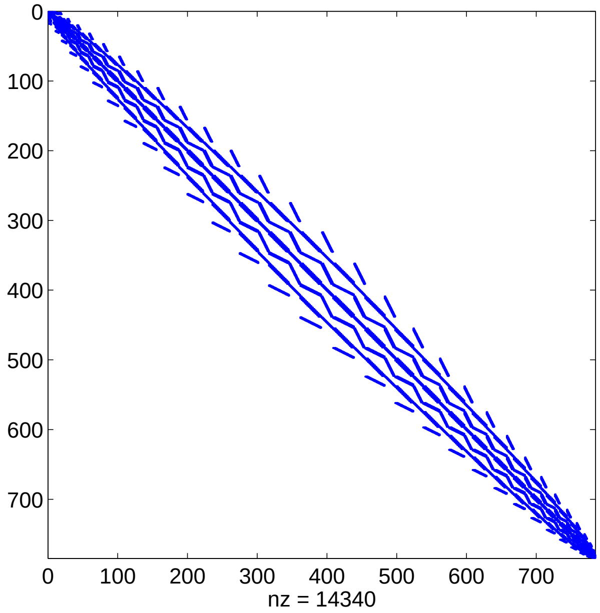


Figure 7.17b: Sparsity pattern of coefficient matrix with Cuthill Mc Kee and p-last/level reordering for 512 elements

7.2.4 Additional Fill-In for the ILU Factorisation Preconditioner

From section 7.2.2 it can be concluded that ILU preconditioning leads to better results for the test model than for example Gauss-Seidel or Eisenstat preconditioning. In section 6.2.2 it is explained how the effect of ILU preconditioning may be improved by allowing some fill-in. Figure 7.18 shows results for ILU preconditioning with additional fill-in together with Cuthill Mc Kee reordering. For convenience, they are compared to results for ILU preconditioning without additional fill-in and Cuthill Mc Kee with p-last/level reordering as well as for the default method, Sloan reordering. The results are remarkable. Up to 32768 elements ($\log_4(n_e/2) = 7$), the number of iterations N increases by a factor 2 if $\log_4(n_e)$ adds 1. For 32768 elements and Cuthill Mc Kee with p-last/level reordering, N is as much as 5 times smaller if some additional fill-in is allowed. On top of that, BiCGstab with ILU preconditioning with additional fill-in and Cuthill Mc Kee with p-last/level reordering does not terminate for 524288 elements ($\log_4(n_e/2) = 9$). Even if the mesh is further refined to 1179648 elements, BiCGstab converges in not more than 3233 iterations. Note that the grow factor of the number of iterations still increases as the mesh is steadily refined. However, this increase is not necessarily excessive, as there is no guarantee that the increase should be linear for mesh refinement and the solver is able to deal properly with large-scaled flow problems.

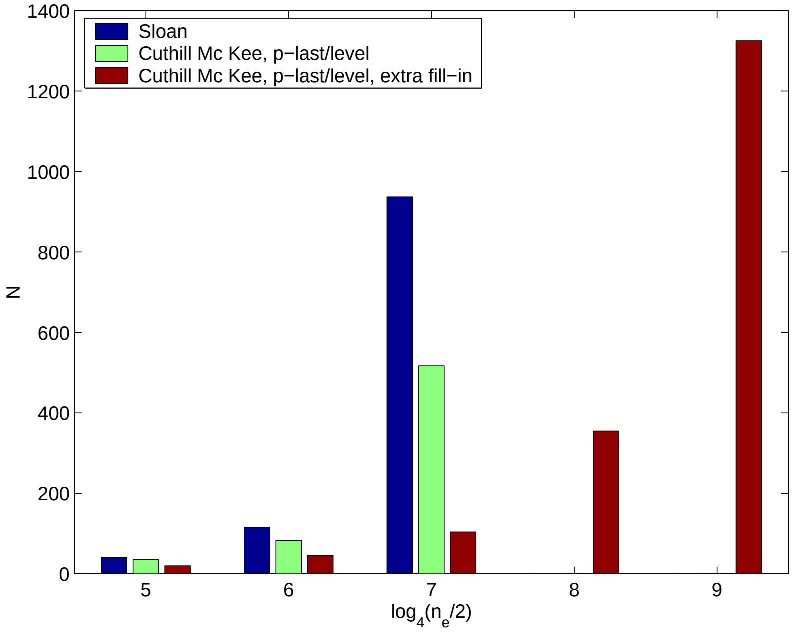


Figure 7.18: Numbers of iterations N for Sloan and Cuthill Mc Kee with p-last/level, with and without additional fill-in for the ILU factorisation preconditioner, for different values of $\log_4(n_e/2)$.

The computational cost of additional fill-in for the ILU preconditioner is known to be considerable [28], Ch. 10. Therefore, it is worth also considering the CPU time. From Table 7.5 it appears that for coarser meshes there is no significant difference between the CPU times for no fill-in and additional fill-in for Cuthill Mc Kee with p-last/level reordering. On the other hand, for 32768 elements, this difference in CPU time is considerable. Indeed, the CPU time per iteration for ILU preconditioning with additional fill is two times larger as without additional fill. However, the improvement in the solver performance is such beneficial that the gain in CPU time is still favourable.

Table 7.5: Number of iterations and CPU time for Sloan and Cuthill Mc Kee with p-last/level, with and without additional fill-in for the ILU preconditioner, for different mesh densities.

			n_e	2048	8192	32768	131072	524288
Sloan		no additional fill	N	41	116	937	-	-
			CPU	2.04E-1	1.77	50.9		
Cuthill Mc Kee	p-last/level	no additional fill	N	35	83	517	-	-
			CPU	1.40E-1	8.65E-1	17.3		
Cuthill Mc Kee	p-last/level	additional fill	N	20	46	104	355	1325
			CPU	1.50E-1	8.45E-1	6.10		

Figure 7.19a-7.20d show the sparsity pattern of ILU for Cuthill MCKee and p-last/level reordering. Figure 7.19a-7.19d illustrate ILU without additional fill, whereas Figure 7.20a-7.20d illustrate ILU with additional fill. It can be observed that for 512 elements, the ILU preconditioner without additional fill has 20834 non-zero entries, whereas the ILU preconditioner with additional fill has 29916 non-zero entries, which is a significant difference. This explains the considerable increase in CPU time per iteration.

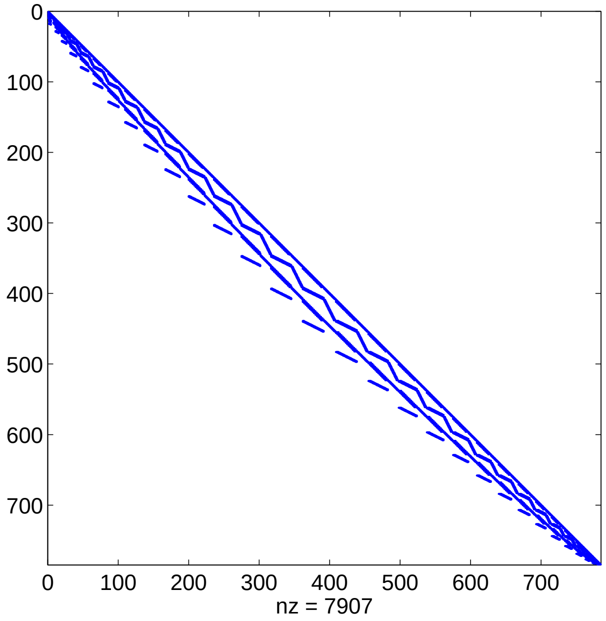


Figure 7.19a: Sparsity pattern of ***L***-matrix in ILU for coefficient matrix with Cuthill Mc Kee and *p*-last/level reordering for 512 elements.

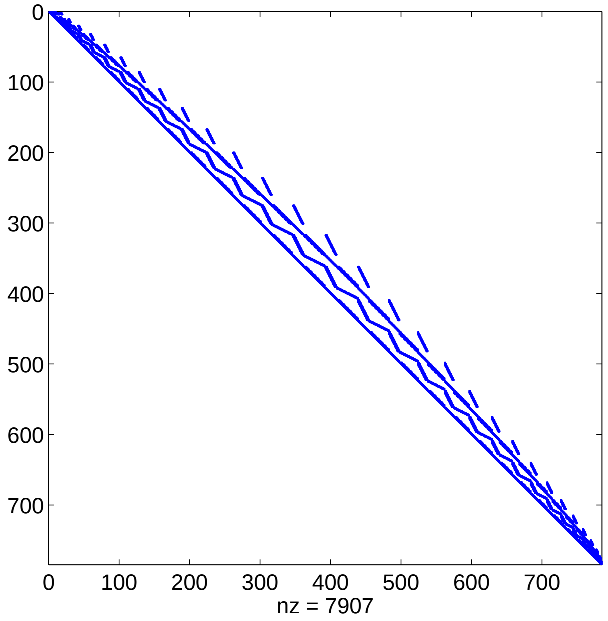


Figure 7.19b: Sparsity pattern of ***U***-matrix in ILU for coefficient matrix with Cuthill Mc Kee and *p*-last/level reordering for 512 elements.

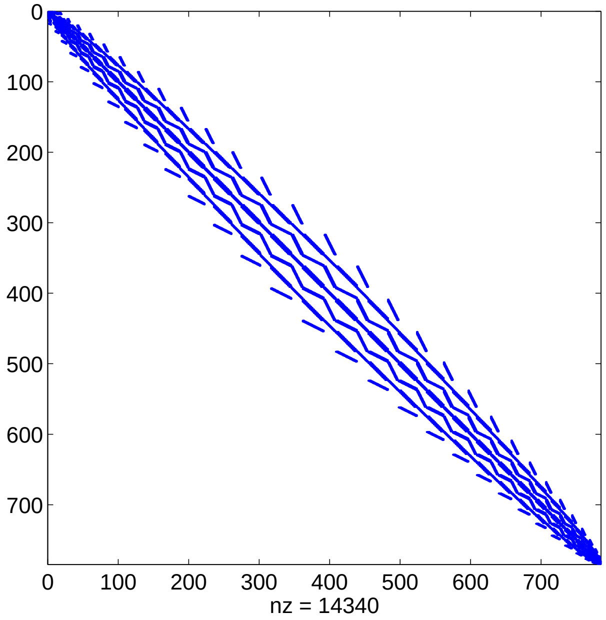


Figure 7.19c: Sparsity pattern of coefficient matrix with Cuthill Mc Kee and *p*-last/level reordering for 512 elements.

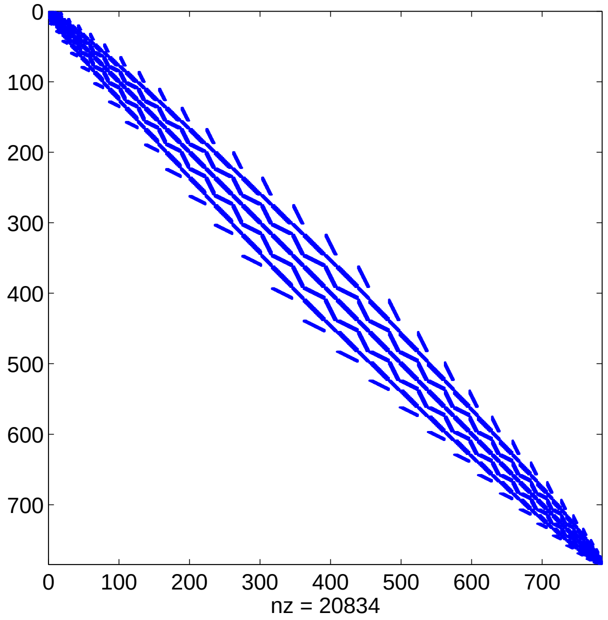


Figure 7.19d: Sparsity pattern of ILU preconditioner for coefficient matrix with Cuthill Mc Kee and *p*-last/level reordering for 512 elements.

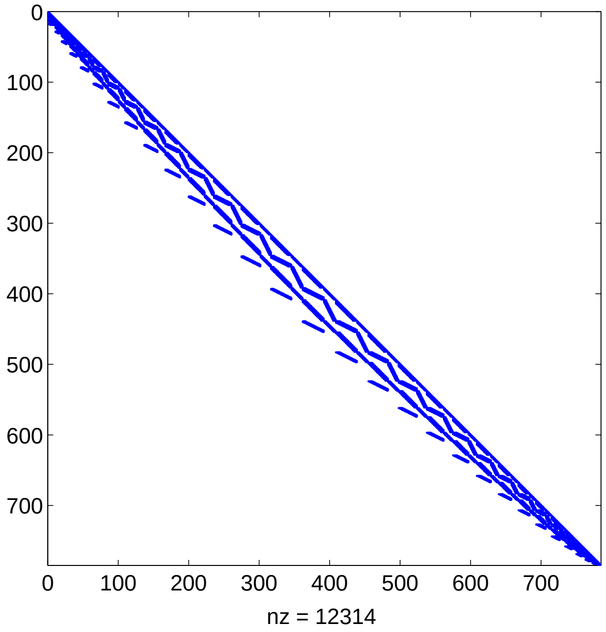


Figure 7.20a: Sparsity pattern of ***L***-matrix in ILU with additional fill for coefficient matrix with Cuthill Mc Kee and *p*-last/level reordering for 512 elements.

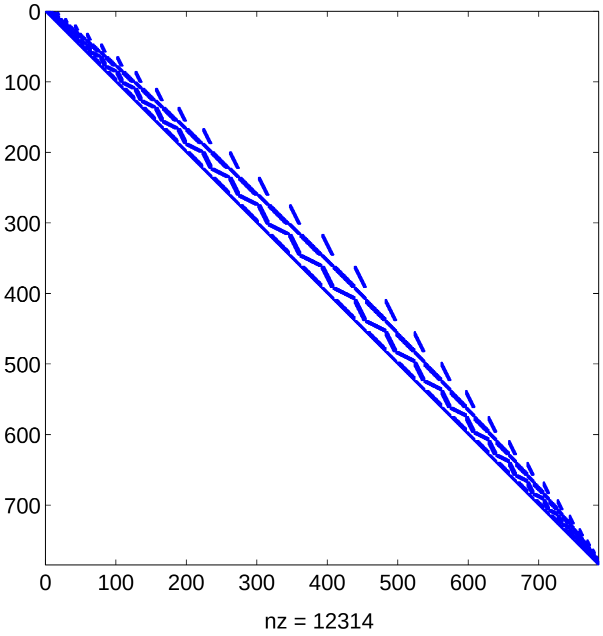


Figure 7.20b: Sparsity pattern of ***U***-matrix in ILU with additional fill for coefficient matrix with Cuthill Mc Kee and *p*-last/level reordering for 512 elements.

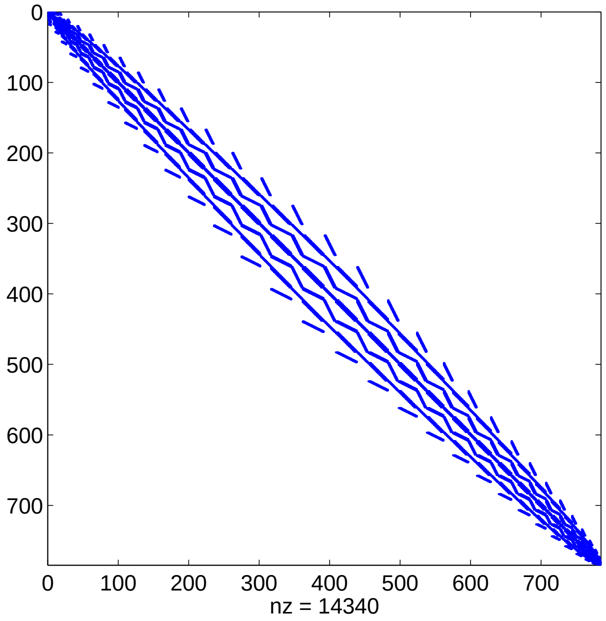


Figure 7.20c: Sparsity pattern of coefficient matrix with Cuthill Mc Kee and *p*-last/level reordering for 512 elements.

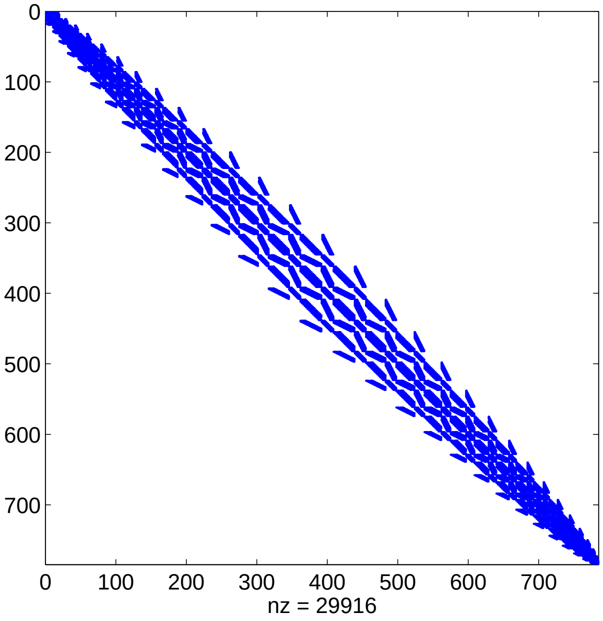


Figure 7.20d: Sparsity pattern of ILU preconditioner with additional fill for coefficient matrix with Cuthill Mc Kee and *p*-last/level reordering for 512 elements.

7.2.5 Different Types of Elements

The solver performance is tested for the stable elements suggested in Chapter 5. Figure 7.21 gives a bargraph with the numbers of iterations for different mesh densities. From this bargraph, it can be seen that elimination of the bubble function at element level gives better results. Furthermore, the solver performance for rectangular elements is better than for triangular elements. By far the best solver performance is observed for rectangular Taylor-Hood elements ($Q2-Q1$). This is contrary to the results for triangular Taylor-Hood elements ($P2-P1$), for which the solver terminates for all mesh densities considered. Although results are best for rectangular elements, they are not preferred because they are less suitable for partitioning curved domains. Of course, isoparametric $Q2-Q1$ elements can be used to fit the curved boundaries, but this may give the additional problem that sign changes in the Jacobian matrix $\nabla_{\xi}\boldsymbol{x}$ may occur due to reversed curved edges of the elements. In addition, note that application of Taylor-Hood elements leads to an increase in the number of unknowns, and hence more computational work is required per iteration. Indeed, this is in contrast to the gain in accuracy. Thus, triangular Mini-elements with elimination of the bubble function at element level are preferred in TNO Glass Group's axi-symmetric pressing model.

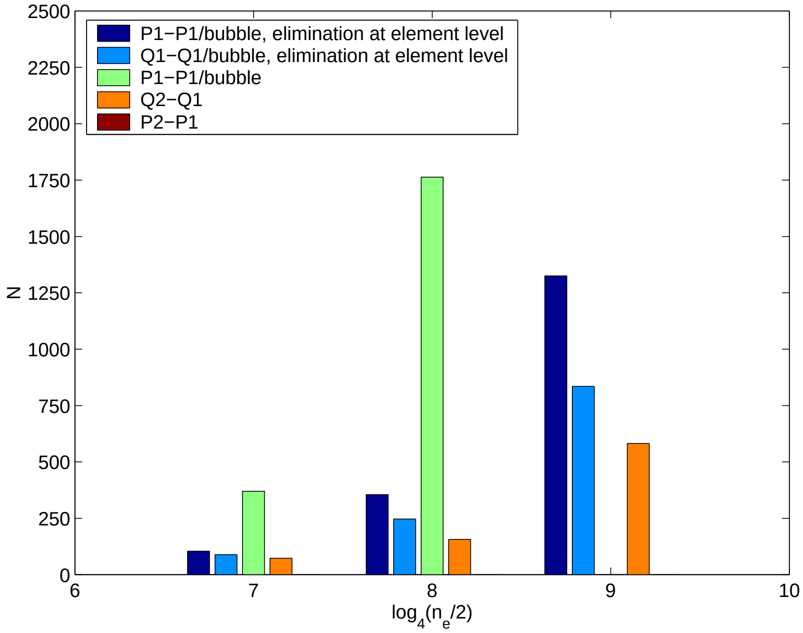


Figure 7.21: Numbers of iterations N for for different elements and for different values of $\log_4(n_e/2)$. The ILU preconditioner with Cuthill Mc Kee with p-last/level reordering and additional fill-in is used. Here, also Mini-elements are considered for which the bubble function is not eliminated at element level.

7.2.6 Conclusions

From the results in this section, the following can be concluded.

- ILU preconditioning leads to better results for the test model than the other preconditioners tested. This preconditioner is also used in TNO Glass Group's pressing model.
- Cuthill Mc Kee reordering of the unknowns gives slightly better results than Sloan reordering. In TNO Glass Group's pressing model, Sloan reordering is used. Furthermore, p-last/level reordering of the unknowns leads to slightly better results.
- Cuthill Mc Kee with p-last/level reordering in combination with additional fill-in for ILU gives substantial improvement of the solver performance.

7.3 Results for an Improved Axi-Symmetrical Glass Pressing Model

The conclusions of section 7.2 indicate that the solver performance for the Stokes flow problem in TNO Glass Group's pressing model can significantly be improved. In this section the test cases for the test model in section 7.2 with positive results are applied to TNO Glass Group's axi-symmetrical pressing model.

7.3.1 Solver Performance for the Stokes Flow Problem

Figure 7.22a-7.22d show results for ILU preconditioning with additional fill and Cuthill Mc Kee with p-last/level reordering applied to TNO Glass Group's axi-symmetrical pressing model. Here, an absolute stop criterium with tolerance $\epsilon = 1\text{E-}3$ is used. In Figure 7.22d ΔN_{900} at the vertical axis denotes the number of iterations for the Stokes flow problem in the 900th time step only. As for the test problem in section 7.2, the improvement is remarkable. However, even for additional fill-in and Cuthill Mc Kee with p-last/level reordering, divergence is observed for $q = 8$ after 900 time steps. Why divergence does not occur at earlier points in time is not precisely clear, but possible causes may be mesh deformation or changes in the flow of glass and air. Also note that in many cases only 1 iteration is required in the 900th time step. It is not likely that this is a consequence of mesh deformation, since it is expected that the solver performance remains the same or becomes worse as the elements are flattened such as illustrated in Figure 5.3d. Apparently, solving the flow problem becomes almost trivial, as the plunger slows down and the flow domain is filled with glass, so that the flow velocity and pressure gradients approach zero.

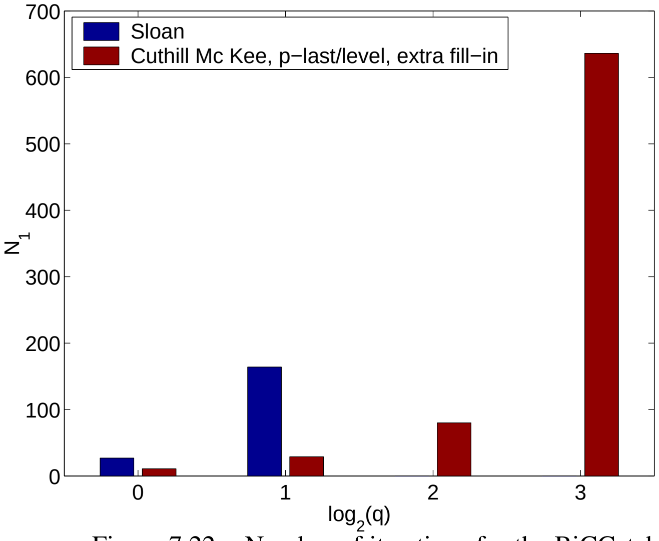


Figure 7.22a: Number of iterations for the BiCGstab solver with ILU preconditioning applied to the discrete Stokes flow problem at $t = 0$ in the pressing model.

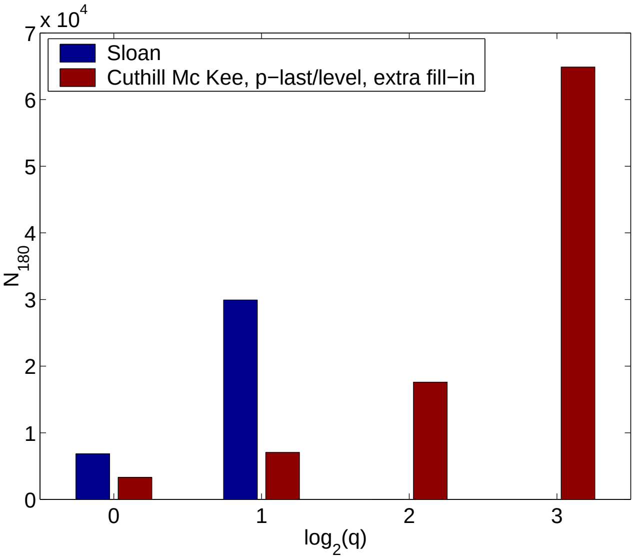


Figure 7.22b: Accumulative number of iterations for the BiCGstab solver with ILU preconditioning at $t = 0.3$ in the pressing model.

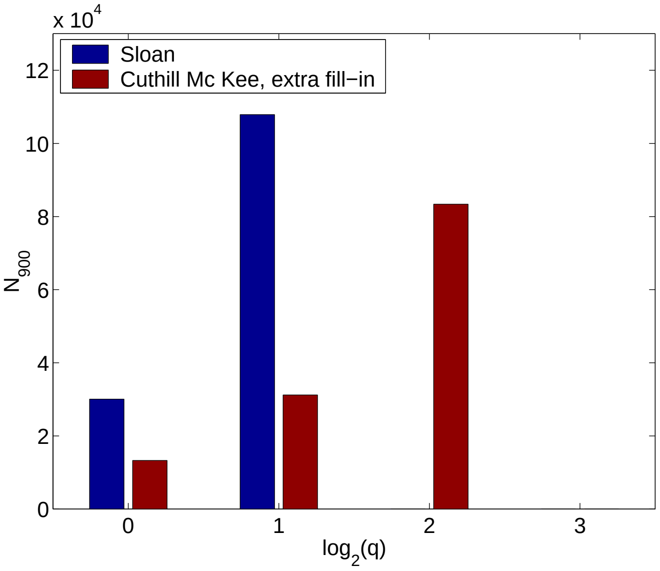


Figure 7.22c: Accumulative number of iterations for the BiCGstab solver with ILU preconditioning at $t = 1.5$ in the pressing model.

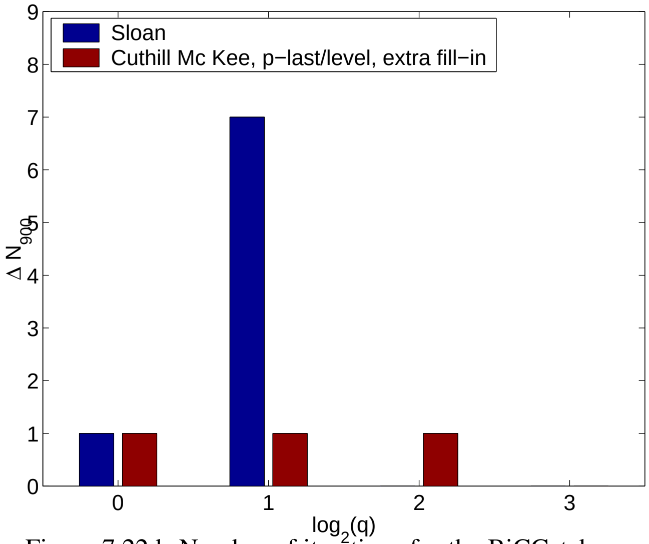


Figure 7.22d: Number of iterations for the BiCGstab solver with ILU preconditioning applied to the discrete Stokes flow problem at $t = 1.5$ in the pressing model.

7.3.2 Numerical Solutions

In this section, numerical solutions of the improved pressing model are examined on accuracy and compared with the solution in section 7.1.2. The flow problems problems are solved by the BiCGstab solver with ILU preconditioning with additional fill and Cuthill Mc Kee with p-last/level ordering. An absolute stop criterium with ϵ = 1E-3 is used.

Figure 7.23a-7.23d show pictures of the flow of glass and air for $q = 1$ at different times. The pictures for $q = 1$ in Figure 7.23a-7.23d are exactly the same as in Figure 7.6a-7.6d in section 7.1.2, from which it may be concluded that the accuracy is the same for both Sloan and Cuthill Mc Kee, p-last/level with extra fill. To obtain a better view of the accuracy of the solutions, the volume conservation of glass is examined. In Figure 7.24a-7.24b, the glass volumes as functions of time for Sloan and Cuthill Mc Kee, p-last level with extra fill are compared for $q = 1$ and $q = 2$, respectively. From these figures it can be concluded that on average the differences in glass volume are insignificant.

Figure 7.25a-7.25f show pictures of the flow of glass and air for $q = 4$ at different times. Since numerical solutions for Sloan and Cuthill Mc Kee, p-last/level with extra fill are almost equal, Figure 7.25a-7.25f can well be compared with the figures in section 7.1.2. From the pictures, it can be observed that the glass-air interface is again much thinner and smoother than for coarser meshes. In comparison with Figure 7.7a-7.7d slight changes in the shape or location of the flow front may be noticed. For example in Figure 7.7d the glass flow has already run onto the far end of the ring in the flow domain, whereas in Figure 7.25f, the flow front hardly can be noticed to touch this part of the boundary. So, the gain in accuracy for mesh refinement from $q = 2$ to $q = 4$ is clearly perceptible in the simulation results.

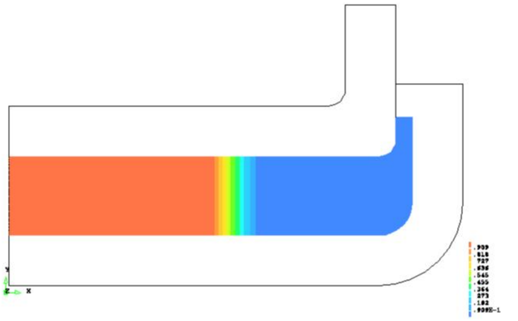


Figure 7.23a: Flow of glass and air in the improved axi-symmetrical pressing model for $q = 1$ at $t = 0$ s. Glass is orange and air is blue.

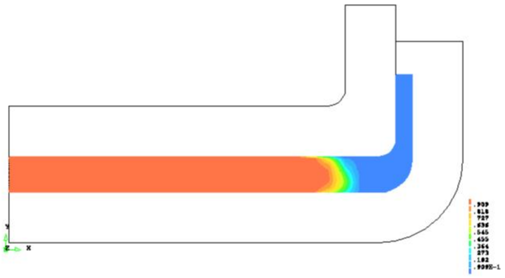


Figure 7.23b: Flow of glass and air in the improved axi-symmetrical pressing model for $q = 1$ at $t = 0.5$ s. Glass is orange and air is blue.

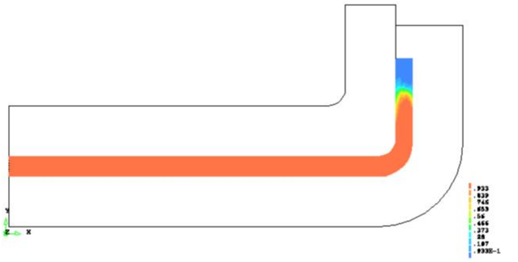


Figure 7.23c: Flow of glass and air in the improved axi-symmetrical pressing model for $q = 1$ at $t = 1.0$ s. Glass is orange and air is blue.

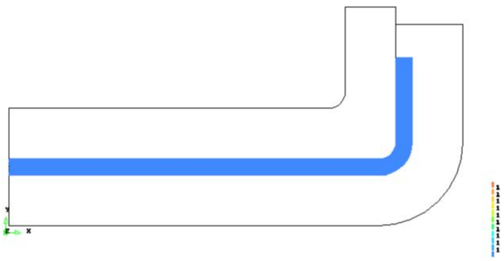


Figure 7.23d: Flow of glass and air in the improved axi-symmetrical pressing model for $q = 1$ at $t = 1.5$ s. Only glass remains.

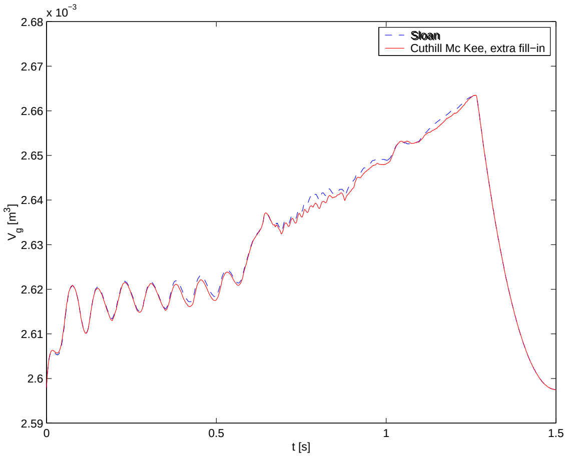


Figure 7.24a: Glass volume (V_g [m^3]) conservation for q = 1.

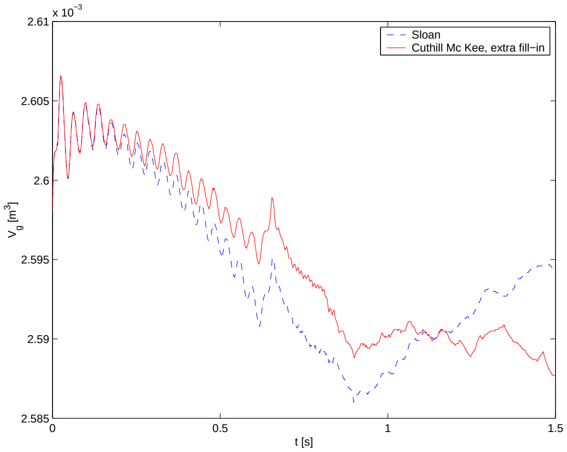


Figure 7.24b: Glass volume (V_g [m^3]) conservation for q = 2.

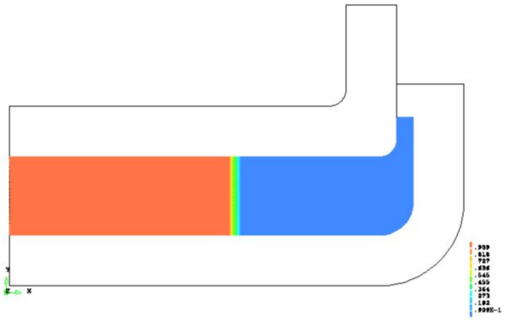


Figure 7.25a: Flow of glass and air in the improved axi-symmetrical pressing model for q = 4 at t = 0 s. Glass is orange and air is blue.

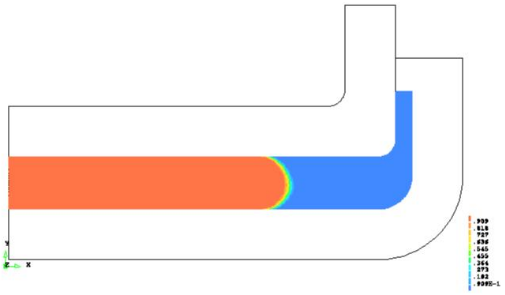


Figure 7.25b: Flow of glass and air in the improved axi-symmetrical pressing model for q = 4 at t = 0.25 s. Glass is orange and air is blue.

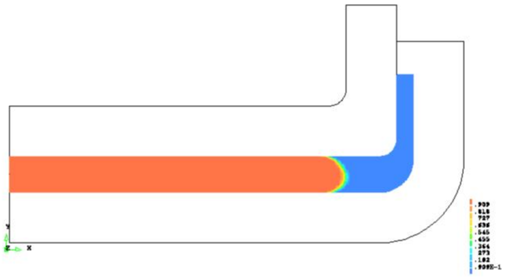


Figure 7.25c: Flow of glass and air in the improved axi-symmetrical pressing model for q = 4 at t = 0.5 s. Glass is orange and air is blue.

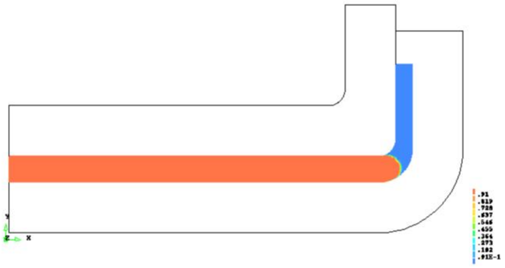


Figure 7.25d: Flow of glass and air in the improved axi-symmetrical pressing model for q = 4 at t = 0.75 s. Glass is orange and air is blue.

Figure 7.25e: Flow of glass and air in the improved axi-symmetrical pressing model for $q = 4$ at $t = 1.0$ s. Glass is orange and air is blue.

Figure 7.25f: Flow of glass and air in the improved axi-symmetrical pressing model for $q = 4$ at $t = 1.5$ s. Glass is orange and air is blue.

Chapter 8

Conclusions and Recommendations

8.1 Conclusions

For the Stokes flow problem in TNO Glass Group's glass pressing process simulation model, problems regarding calculation speed of iterative solvers occur, as application of mesh refinement produces an excessive increase in the number of iterations, and finally the solver even terminates. This solver problem can also be seen in a simple test model for the simulation of a pressing process time step. Although the spectral condition number in the pressing model is substantial, only small condition numbers are obtained in the test problems.

A thorough examination of the discretisation methods for Stokes flow problems in TNO Glass Group's axi-symmetrical pressing model leads to the conclusion that they are stable and result in well-defined linear systems of equations. However, since the behaviour of iterative solvers is in general rather unpredictable, this gives no guarantee that the solver performance will be reliable.

For the finite element discretisation triangular Mini-elements are used. The bubble functions are eliminated by means of static condensation at element level. The elements are proven to be stable. Triangular Mini-elements are preferable above the other elements that have been tested, because the solver performance for triangular Mini-elements is satisfactory in comparison to the other elements, Mini-elements require low computational cost and triangular elements are more appropriate for a partition of curved domains.

Preconditioning offers good prospects to improve the solver performance. Preconditioners tested are Gauss-Seidel, Eisenstat and ILU. Out of these, ILU gives the best results. Some improvement in the number of iterations is gained by applying Cuthill Mc Kee reordering instead of Sloan reordering. Substantial improvement in the solver performance can be observed if additional fill for the ILU is allowed. Although additional fill for ILU results in a significant increase in CPU time per iteration, the gain in efficiency is still considerable. Most importantly, after additional fill for the ILU is applied, the discrete Stokes flow problem can be solved by means of an iterative solver on relatively fine meshes, for which the solvers terminated before.

Although the improvement gained by ILU with additional fill is encouraging, iterative solvers for the pressing model still terminate on extremely fine meshes as the plunger reaches it lowest position. Possible causes of this are the mesh deformation in the pressing model, during which the elements are flattened as the plunger moves down, or changes in the flow of glass and air. The mesh deformation may have a subordinate influence on the solver performance.

8.2 Recommendations

The solver performance still leaves room for improvement. Firstly, the influence of mesh deformation on the solver performance can be studied into more detail. A mesh adaption algorithm that avoid mesh deformation may be applied to overcome termination of the iterative solver, but existing mesh algorithms are known to be rather inefficient and therefore cannot be recommended for general use. Some improvement may also be gained by taking elements of different thickness. However, it is not clear how much improvement can be gained by avoiding too much flattening of the elements. Secondly, different preconditioners can be used. Notably, multigrid methods are highly recommended in literature. For example, M. F. Adams et al [1] states that he can solve a solid mechanics problem with up to 237 million degrees of freedom with an algebraic multigrid method as preconditioner for a conjugate gradient solver. However, ILU preconditioning with Cuthill Mc Kee reordering and additional fill already results in a positive solver performance, and it is doubtful how much improvement can be gained by application of different preconditioners. Since results of ILU with additional fill are already encouraging, it may as well be a good idea to test different incomplete factorisation preconditioners, such as ILUT [28]. Note that preconditioners such as multigrid methods and ILUT have yet to be implemented in Sepran, which may be quite time consuming and lead to additional difficulties.

The ILU preconditioner with Cuthill Mc Kee reordering and additional fill has not (yet) been applied to TNO Glass Group's three dimensional pressing model. The preconditioner may not necessarily have the same effect on the three dimensional pressing model as on the axi-symmetric pressing model. In the three dimensional model the profile or the bandwidth of the coefficient matrix may just become to large by allowing some additional fill, so that hardly any or no improvement in the solver performance will be gained. Nonetheless, regarding the considerable improvement in the axi-symmetric model, application of the preconditioner to the three dimensional problem is quite promising.

Appendix A

Vector Spaces

This appendix is devoted to vector spaces and their corresponding inner products that arise from functional analysis. The vector spaces are defined in association with a polygonal bounded domain $\Omega \in \mathbb{R}^n$.

A.1 The Vector Space of Lebesgue p-Integrable Functions

Let $p \in \mathbb{N}$. Define the set of Lebesgue p-integrable functions as

$$L^p(\Omega) = \Big\{u : \Omega \to \mathbb{R} \;\Big|\; \int_\Omega |u|^p \mathrm{d}\boldsymbol{x} < \infty\Big\}. \tag{A.1.1}$$

The related $L^p(\Omega)$ norm is

$$||u||_{L^p(\Omega)} = \left(\int_\Omega |u|^p \mathrm{d}\boldsymbol{x}\right)^{\frac{1}{p}}. \tag{A.1.2}$$

A special case is the set of Lebesgue square integrable functions,

$$L^2(\Omega) = \Big\{u : \Omega \to \mathbb{R} \;\Big|\; \int_\Omega u^2 \mathrm{d}\boldsymbol{x} < \infty\Big\}. \tag{A.1.3}$$

For $u, v \in L^2(\Omega)$, the $L^2(\Omega)$ inner-product is given by

$$(u, v)_{L^2(\Omega)} = \int_\Omega u\, v\, \mathrm{d}\boldsymbol{x}. \tag{A.1.4}$$

It can be seen that the induced norm, $||u||_{L^2(\Omega)} = \sqrt{(u, u)_{L^2(\Omega)}}$ conforms to definition (A.1.2).

A.2 The Vector Space of Essentially Bounded Measurable Functions

Define the set of essentially bounded measurable functions as

$$L^\infty(\Omega) = \left\{ u : \Omega \to \mathbb{R} \,\middle|\, \|u\|_\infty < \infty \right\}, \tag{A.2.1}$$

where the $L^\infty(\Omega)$ norm is set to

$$\|u\|_\infty = \inf \left\{ C \geq 0 \,\middle|\, |u| \leq C \text{ almost everywhere on } \Omega \right\}. \tag{A.2.2}$$

A.3 Sobolev Spaces

The Sobolev space $H^1(\Omega)$ is defined by

$$H^1(\Omega) = \left\{ u \in L^2(\Omega) \,\middle|\, \int_\Omega |\nabla u|^2 \mathrm{d}\boldsymbol{x} < \infty \right\}. \tag{A.3.1}$$

For $u, v \in H^1(\Omega)$, the $H^1(\Omega)$ inner-product is given by

$$(u, v)_{H^1(\Omega)} = \int_\Omega (\nabla u \nabla v + u\, v) \mathrm{d}\boldsymbol{x}. \tag{A.3.2}$$

The $H^1(\Omega)$ inner-product induces a $H^1(\Omega)$ norm:

$$\|u\|_{H^1(\Omega)} \;=\; \sqrt{(u, u)_{H^1(\Omega)}} \;=\; \left(\int_\Omega (|\nabla u|^2 + |u|^2) \mathrm{d}\boldsymbol{x} \right)^{\frac{1}{2}}. \tag{A.3.3}$$

In addition, the constrained Sobolev space is

$$H^1_g(\Omega; \Gamma) = \left\{ u \in H^1(\Omega) \,\middle|\, u\Big|_\Gamma = g \right\}, \tag{A.3.4}$$

for some subset $\Gamma \subset \Omega$ and $g : \Gamma \mapsto \mathbb{R}$. In view of weak formulations, it is important to note that $C^2(\Omega)$ is dense in $H^1(\Omega)$ [7].

Appendix B

Analysis of the Behaviour of Flow Solutions at the Glass-Air Interface near the Equipment Boundary

This appendix focusses on the possibility of numerical instability due to singular behaviour of flow solutions at the glass-air interface in a neighbourhood of the equipment boundary. Consider a point P on $\Gamma_i \cap \Gamma_e$, with $\Gamma_e = \Gamma_m \cup \Gamma_r \cup \Gamma_p$ (see Figure B.1). Consider a small neighbourhood Ω_P of P in Ω. Assume constant viscosity in Ω_P. So, the Stokes flow equations read

$$\mu \Delta \boldsymbol{u} - \nabla p = 0, \tag{B.1}$$

$$\nabla \cdot \boldsymbol{u} = 0. \tag{B.2}$$

Consider the rotation of the momentum equations around P. Then since

$$\nabla \times \nabla p = 0,$$

the flow velocity satisfies

$$\nabla \times \Delta \boldsymbol{u} = \boldsymbol{0}, \qquad \text{in } \Omega_P. \tag{B.3}$$

As the flow problem is not essentially three-dimensional and only a small vicinity of a point on the equipment boundary is considered, it is convenient to simplify the flow problem to two-dimensional Cartesian coordinates:

$$\boldsymbol{u} = u\boldsymbol{e}_x + w\boldsymbol{e}_z, \qquad \text{in } \Omega_P. \tag{B.4}$$

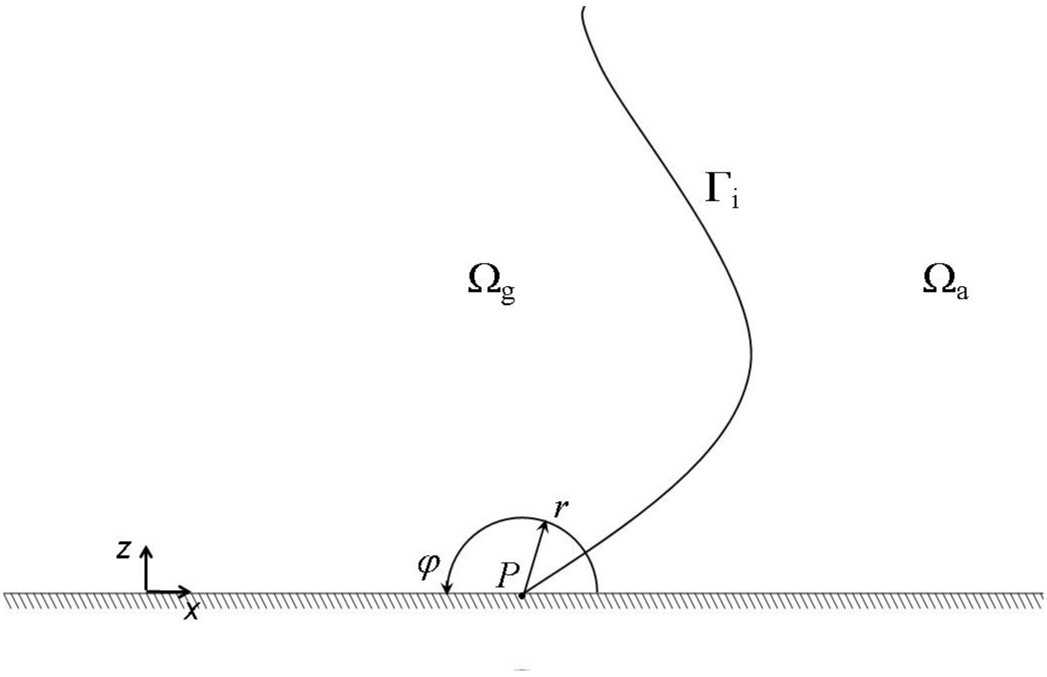


Figure B.1: Neighbourhood of equipment boundary of glass-air-interface.

Then as a result of the continuity equation, the flow velocity can be written in terms of a stream function ψ,

$$\boldsymbol{u} = \nabla \times \psi \boldsymbol{e}_y, \qquad \text{in}\, \Omega_P. \tag{B.5}$$

In terms of (B.5), only the y-component of (B.3) is non-trivial and results in the bi-harmonic equation

$$\Delta^2 \psi = 0, \qquad \text{in}\, \Omega_P. \tag{B.6}$$

Since P is the centre of the domain of interest, it is convenient to write the stream function ψ locally in terms of polar coordinates (r, θ), where $r = \sqrt{(x - x_P)^2 + (z - z_P)^2}$ (see Figure B.1). To this end, consider a semicircle $\mathbb{B}_{P,\,\delta} \subset \Omega_P$ around P for some $\delta > 0$, given by

$$\mathbb{B}_{P,\,\delta} = \Big\{(r, \theta)\Big| 0 < r < \delta,\ \theta_0 \leq \theta \leq \pi\Big\}, \tag{B.7}$$

where θ_0 is the angle of the glass air interface to the equipment boundary of the air domain. So, the semicircle is bounded by the equipment wall, the glass-air interface and a maximum distance $r = \delta$. In order to impose BCs on the equipment wall, it should be noticed that the glass-air interface is propagating to the right in time. Without loss of generality, assume that the interface at point P is propagating with constant velocity $\boldsymbol{u}_P = (1,0)^T$ for a short period of time. Next, consider $\boldsymbol{u}$ as the velocity with respect to the interface, that is $\boldsymbol{u} := \boldsymbol{u} - \boldsymbol{u}_P$. Then the no-slip condition for glass induces

$$u\Big|_{\phi=\pi} = -1. \tag{B.8}$$

So, relatively to the interface, each point on the equipment boundary is moving to the left. In addition, adopt the BC for an impenetrable wall

$$v\Big|_{\phi=\pi} = 0. \tag{B.9}$$

In order to find a solution of (B.6), separation of variables with respect to (r,θ) is applied:

$$\psi(r,\theta) = R(r)\Theta(\theta). \tag{B.10}$$

This seems reasonable, since any external contributions to the flow are omitted and the semicircular flow domain can basically be considered infinitely extended. In terms of (B.10), the flow velocity components can be written as

$$u(r,\theta) = \psi_x(r,\theta) \;=\; R'(r)\Theta(\theta)\cos\theta + r^{-1}R(r)\Theta'(\theta)\sin\theta, \tag{B.11}$$

$$v(r,\theta) = -\psi_y(r,\theta) = -R'(r)\Theta(\theta)\sin\theta + r^{-1}R(r)\Theta'(\theta)\cos\theta. \tag{B.12}$$

These solutions should satisfy BCs (B.8)-(B.9): BC (B.8) gives

$$\Theta(\pi) = \frac{1}{R'}, \tag{B.13}$$

and BC (B.9) leads to

$$\Theta'(\pi) = 0. \tag{B.14}$$

From (B.13), it follows that $R' = R_1$ is constant, so that R is linear:

$$R(r) = R_1 r + R_0. \tag{B.15}$$

As a result, substitution of (B.10) into PDE (B.6) yields

$$rR_1(\Theta'''' + 2\Theta'' + \Theta) + R_0(4\Theta'''' + \Theta'') = 0, \tag{B.16}$$

or

$$R_1(\Theta'''' + 2\Theta'' + \Theta) = 0, \tag{B.17}$$

$$R_0(4\Theta'''' + \Theta'') = 0. \tag{B.18}$$

Since $R_1 \neq 0$, it holds that

$$\Theta'''' + 2\Theta'' + \Theta = 0, \tag{B.19}$$

which has a solution of the form

$$\Theta(\theta) = Ae^{i\theta} + B\theta e^{i\theta} + Ce^{-i\theta} + D\theta e^{-i\theta}, \tag{B.20}$$

for some constants A, B, C, D. Since (B.20) is not a solution of $4\Theta'''' + \Theta''$, it follows from (B.18) that $R_0 = 0$. Without loss of generality, choose $R_1 = 1$. It remains to find the coefficients in (B.20). Coefficients A and C can be determined from BCs (B.13)-(B.14):

$$A = -\Big(B\pi + \frac{i}{2}(D - B) + \frac{1}{2}\Big), \tag{B.21}$$
$$C = -\Big(D\pi - \frac{i}{2}(D - B) + \frac{1}{2}\Big). \tag{B.22}$$

Next, in order to have a real flow velocity, it is required that the imaginary part of Θ is zero. This results in the system of equations

$$\begin{cases} \mathrm{Im(B)} - \mathrm{Im(D)} - \mathrm{Re(B)}\pi + \mathrm{Re(D)}\pi = 0, \\ \mathrm{Re(B)} - \mathrm{Re(D)} = 0, \\ -\mathrm{Im(B)}\pi - \mathrm{Im(D)}\pi = 0, \\ \mathrm{Im(B)} + \mathrm{Im(D)} = 0. \end{cases} \tag{B.23}$$

This system has solution $B = D \in \mathbb{R}$. Finally, it follows that

$$R(r) = r, \tag{B.24}$$
$$\Theta(\theta) = \Big(2D(\theta - \pi) - 1\Big)\cos\theta. \tag{B.25}$$

Substitution of (B.24) and (B.25) into flow velocity (B.11)-(B.12) gives flow solution

$$u(\theta) = \Big(2D(\theta - \pi) - 1\Big)\cos 2\theta + D\sin 2\theta, \tag{B.26}$$
$$v(\theta) = -\Big(2D(\theta - \pi) - 1\Big)\sin 2\theta + D(\cos 2\theta + 1). \tag{B.27}$$

Since the flow solution (B.26)-(B.27) is independent of the radius, point P is not a singular point of this solution. The remaining constant D depends on the flow of air in a vicinity of P. If for simplicity air is replaced by vacuum in the flow calculations, it can be assumed that the velocity gradient is zero at the interface. This involves the additional BCs:

$$\boldsymbol{n} \cdot \nabla \boldsymbol{u} \cdot \boldsymbol{n}\Big|_{\theta=\theta_0} = 0, \tag{B.28}$$
$$\boldsymbol{n} \cdot \nabla \boldsymbol{u} \cdot \boldsymbol{t}\Big|_{\theta=\theta_0} = 0, \tag{B.29}$$

where

$$\nabla \boldsymbol{u} = \frac{1}{r}\begin{pmatrix} 0 & \frac{1}{2}(\partial_\theta u - v) \\ \frac{1}{2}(\partial_\theta u - v) & \partial_\theta v - u \end{pmatrix}, \tag{B.30}$$

$$\boldsymbol{n} = \begin{pmatrix} \cos\theta_0 \\ \sin\theta_0 \end{pmatrix}, \tag{B.31}$$

$$\boldsymbol{t} = \begin{pmatrix} -\sin\theta_0 \\ \cos\theta_0 \end{pmatrix}. \tag{B.32}$$

Application of BCs (B.28)-(B.29) results in $D = 0$ and $\theta_0 = 0$. Apparently, for the given assumptions there is no solution with $\theta_0 > 0$. However, in view of the round flow front in the glass pressing process simulation results in Chapter 7, $\theta_0 = 0$ seems to be a satisfactory approximation of the exact angle. The corresponding flow velocity field is

$$\boldsymbol{u}(\theta) = \begin{pmatrix} -\cos 2\theta \\ \sin 2\theta \end{pmatrix}, \qquad 0 < \theta \leq \pi. \tag{B.33}$$

Figure B.2 shows the vector field corresponding to the flow velocity relatively to the equipment boundary. The flow velocity properly follows the propagation of the glass-air interface and no suspicious behaviour can be encountered. So, the analytic results are in agreement with the numerical experiments.

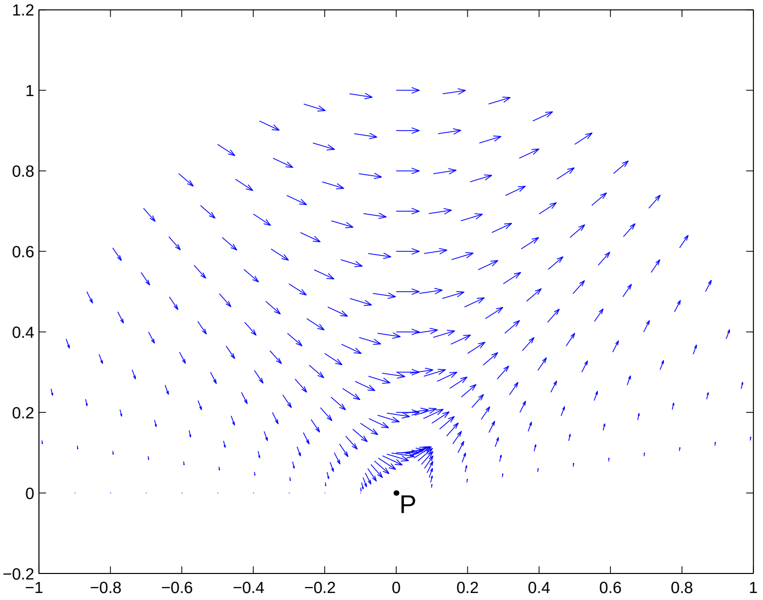


Figure B.2: Vector field of flow solution in a small vicinity of point P at the equipment boundary between glass and vacuum.

Bibliography

[1] M. F. Adams, H. Bayraktar, T. Keaveny, and P. Papadopoulos. Applications of algebraic multigrid to large-scale finite element analysis of whole bone micro-mechanics on the ibm sp. In ACM/IEEE Proceedings of SC2003: High Performance Networking and Computing, 2003.

[2] O. Axelson and V. A. Barker, editors. Finite Element Solutions of Boundary Value Problems: Theory and Computation. Academic Press, London, 1984.

[3] F. P. T. Baaijens. Applied Computational Mechanics. Eindhoven University of Technology, 1991.

[4] M. Benzi, D. B. Szyld, and A. van Duin. Orderings for incomplete factorization preconditioning of nonsymmetric problems. Siam J. Sci. Comput., 20(5):1652–1670, 1999.

[5] F. Brezzi. On the existence, uniqueness and approximation of saddlepoint problems arising from lagrangian multipliers. RAIRO Anal. Numer., 8:129–151, 1974.

[6] F. Brezzi and R. S. Falk. Stability of higher-order hood-taylor methods. SIAM J. Numer. Anal., 28(3):581–590, 1991.

[7] V. I. Burenkov. Sobolev Spaces on Domains. B. G. Teubner, Stuttgart-Leipzig, 1998.

[8] Z. Chen. Finite Element Methods and Their Applications. Springer, Berlin, 2005.

[9] C. Cuvelier, A. Segal, and A. A. Steenhoven. Finite Element Methods and Navier-Stokes Equations. Reidel, Dordrecht, 1986.

[10] C. C. Douglas. Multigrid methods in science and engineering. IEEE Computational Science and Engineering, 3(4):55–68, 1996.

[11] L. C. Evans. Partial Differential Equations. American Mathematical Society, Providence, Rhode Island, 1998.

[12] B. Fischer. Polynomial Based Iteration Methods for Symmetric Linear Systems. Wiley, Chichester, 1996.

[13] C. G. Giannopapa. Development of a computer simulation model for blowing glass containers. In ASME Proceedings of PVP2006: Pressure Vessels and Piping Division Conference, 2006.

[14] V. Girault and P. A. Raviart. Finite Element Methods for Navier-Stokes Equations: Theory and Algorithms. Springer, Berlin, 1986.

[15] J. van Kan, G. Segal, and F. Vermolen. Numerical Methods in Scientific Computing. VSSD, Delft, 2005.

[16] D. Krause and H. Loch, editors. Mathematical Simulation in Glass Technology. Springer, Berlin, 2002.

[17] K. Laevsky. Pressing of Glass in Bottle and Jar Manufacturing: Numerical Analysis and Computation. PhD thesis, Eindhoven University of Technology, 2003.

[18] J. E. Maitre and E. Wabo. Stabilized formulations and mini-element for the n-dimensional stokes equations: Properties and solution procedure. In M. Kriek, P. Neittaanmki, and R. Stenberg, editors, Finite Element Methods: 50 Years of the Courant Element, pages 285–299, New York, 1994. Marcel Dekker.

[19] J. M. L. Maubach. The Finite Element Method. Eindhoven University of Technology, 2006.

[20] T. M. Nagtegaal. Optical Method for Temperature Profile Measurements in Glass Melts. PhD thesis, Eindhoven University of Technology, 2002.

[21] A. Ouazzi. Finite Element Simulation of Nonlinear Fluids with Application to Granular Material and Powder. PhD thesis, University of Dortmund, 2005.

[22] A. Padiy. Reliable Iterative Methods for Solving Ill-Conditioned Algebraic Systems. PhD thesis, Catholic University of Nijmegen, 2000.

[23] R. Pierre. Simple c^0-approximations for the computation of incompressible flows. Comput. Methods Appl. Mech. Engrg., 68:205–228, 1988.

[24] G. Prokert. Variational Methods and Elliptic Equations. Eindhoven University of Technology, 2004.

[25] J. N. Reddy and D. K. Gartling. The Finite Element Method in Heat Transfer and Fluid Dynamics. CRC Press LCL, New York, second edition, 2001.

[26] M. Rehman, C. Vuik, and G. Segal. Solution of the incompressible navier stokes equations with preconditioned krylov subspace methods. Technical report, Delft University of Technology, 2006.

[27] S. W. Rienstra and Chandra T. D. Analytical approximations to the viscous glass-flow problem in the mould-plunger pressing process, including an investigation of boundary conditions. Journal of Engineering Mathematics, 39:241–259, 2001.

[28] Y. Saad. Iterative Methods for Sparse Linear Systems. PWS Publishing, London, 1996.

[29] A. Segal. Sepran Introduction. Ingenieursbureau SEPRA, 1993.

[30] A. Segal. Sepran Standard Problems. Ingenieursbureau SEPRA, 1993.

[31] A. Segal. Sepran User Manual. Ingenieursbureau SEPRA, 1993.

[32] G. Segal and K. Vuik. A simple iterative linear solver for the 3d incompressible navier-stokes equations disretized by the finite element method. Technical report, Delft University of Technology, 1995.

[33] J. A. Sethian. Level Set Methods and Fast Marching Methods. Cambridge University Press, USA, 1999.

[34] J. E. Shelby. Introduction to Glass Science and Technology. The Royal Society of Chemistry, Cambridge, second edition, 2005.

[35] J. R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. Technical report CMU-CS-94-125, Carnegie Mellon University, 1994.

[36] D. Silvester and A. Wathen. Fast iterative solution of stabilised stokes systems part ii: Using general block preconditioners. SIAM J. Numer. Anal., 31(5):1352–1367, 1994.

[37] H. A. van der Vorst. Bi-cgstab: a fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems. SIAM J. Sci. Stat. comput., 13(2):631–644, 1992.

[38] H. de Waal and R. G. C. Beerkens, editors. NCNG Handboek voor de Glasfabricage. TNO-TPD-Glastechnologie, second edition, 1997.

[39] U. M. Yang. Preconditioned conjugate gradient-like methods for nonsymmetric linear systems. CSRD-report, no. 1210, University of Illinois at Urbana-Champaign, 1994.