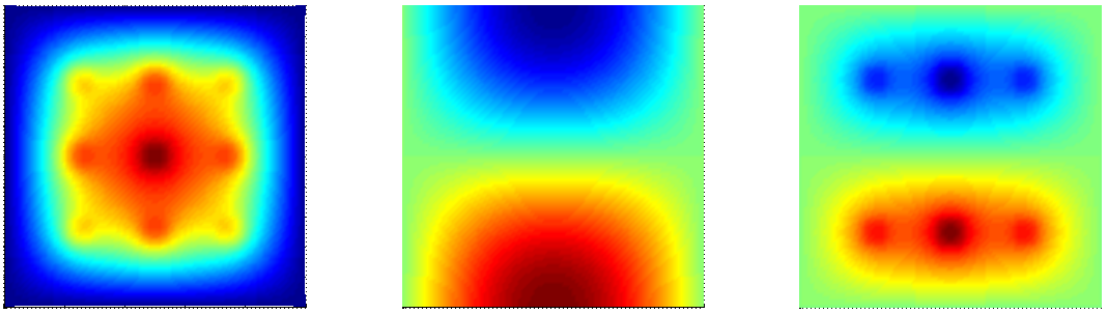
Technische Universität Berlin
Institut für Mathematik

Application of Finite Volume and Finite Element methods to distributed optimal control of semi-linear elliptic equations

Master's Thesis

in Computer Simulations for Science and Engineering



Author:
Giedrius Arbaciauskas

Supervisors:
Prof. Dr. Fredi Tröltzsch
Prof. Dr. Reinhard Nabben

March 2015

Declaration of Authorship

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstsändig und eigenhändig sowie ohne unerlaubte fremde Hilfe und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

Berlin, den

.....
Unterschrift

Acknowledgements

This thesis would not have been possible without the support of many people. First of all, I wish to thank my supervisor, professor Dr. Fredi Tröltzsch, for his assistance and guidance throughout this work. Also, I would like to thank my family, especialy my mother for the financial and moral support during this time. I would also like to thank my program leader Dr. Reinhard Nabben for his support during the year in Berlin.

TECHNISCHE UNIVERSITÄT BERLIN
Institut für Mathematik

Zusammenfassung

Master of Computer Simulations for Science and Engineering

Application of Finite Volume and Finite Element methods to distributed optimal control of semi-linear elliptic equations

by Giedrius Arbaciauskas

Diese Arbeit befasst sich mit dem Problem der optimalen Steuerung linearer sowie semilinearer partieller Differentialgleichungen zweiter Ordnung, wobei die Steuerung über das Gebiet Ω verteilt ist. Das semilineare elliptische Randwertproblem wird analysiert, indem sowohl Existenz einer eindeutigen Lösung und einer optimalen Steuerung gezeigt, als auch notwendige Optimalitätsbedingungen hergeleitet werden. Die Probleme werden unter Verwendung des *Finite-Volumen-Verfahrens* und der *Finite-Elemente-Methode* diskretisiert. Der Semilinearität wird dadurch begegnet, dass die Differentialgleichung mithilfe des Newton-Verfahrens linearisiert wird. Das Ziel dieser Arbeit ist es, verschiedene Optimierungsverfahren und Diskretisierungstechniken vorzustellen und anzuwenden, um den Zustand und die Steuerung zu finden, welche das dazugehörige Kostenfunktional unter Erfüllung linearer und semilinearer partieller Differentialgleichungen als Nebenbedingung minimieren.

Beim ersten Optimierungsverfahren wird das lineare Optimalsteuerungsproblem auf ein quadratisches Optimierungsproblem reduziert, welches in MATLAB mittels *quadprog* gelöst werden kann. Bei der zweiten Methode werden beide Probleme mithilfe von *fmincon* optimiert, einmal mit und einmal ohne Verwendung des Gradienten des reduzierten Kostenfunktionals. Der Vergleich der Resultate ergibt, dass ersteres zu einer erheblichen Laufzeitverminderung führt, da der Gradient bereits im Vorfeld berechnet wird. Schließlich wird die Methode der projizierten Gradienten eingeführt, welche von allen Verfahren die größte Effizienz aufweist.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Optimal control is of great importance in many areas of science such as aerospace, robotics, economics, chemistry, mechanics, physics, vehicle dynamics ,life sciences and in many other fields. Processes to be optimized can be modelled by *ordinary differential equations* (ODEs) [1–3], where only one independent variable is used, however not everything can be modelled in this way. Therefore, *partial differential equations* (PDEs) need to be used, in which two or more independent variables appear [4–10]. Some examples would be heating, cooling and fluid flow problems. Thus, optimal control of PDEs is described in this thesis. In many cases we want to control these processes to achieve a desired state. Therefore, the optimal control theory is required [5, 11, 12].

Optimal control problems are formulated as optimization problems governed by a PDE, sometimes constrained on a control. There are many types of PDEs. In this thesis we focus on second-order linear and semi-linear elliptic equations [5, 13, 14]. A discretization of the problem has to be done as the equations are infinite dimensional equations. Two advanced numerical methods, *Finite Volume Method* (FVM) [4, 15] and *Finite Element Method* (FEM) [4–6, 13–16] will be introduced and used to obtain the discretization. However the discretization must be very fine in order to achieve more accurate results.

After the problem is transformed into optimization problem and discretized, it can be solved using many different optimization methods. Popular mathematical modelling program *MATLAB* is used for this purpose, which enables us to solve and visually represent different optimal control problems [16, 17]. Optimization functions such as quadprog, fmincon and fmincon with supplied gradient as well as projected gradient method are used to minimize the problem.

The objective of this thesis is to present and discretize linear and semi-linear elliptic distributed control problems using different numerical methods. Then transform optimal

control problem into optimization problem, solve it using different optimization methods and compare the results.

At first the problem is introduced in Chapter 2. Then the existence of a solution and necessary optimality conditions are discussed.

In Chapter 3 and 4 two numerical methods are introduced: *Finite Volume Method* (FVM) and *Finite Element Method* (FEM). Then the optimal control problem is discretized.

Optimal control problem is transformed into optimization problem in Chapter 5 and several optimization methods are introduced. Finally, the numerical results of three-dimensional examples are presented and different methods are compared, followed by the conclusion in Chapter 6.

Chapter 2

Distributed optimal control of elliptic PDE

The optimal control of PDEs is an optimization problem. A state on a space domain is considered, given by the PDE. In many cases we want to control this state in order to achieve a desired solution. For example, steel factory produces thick plates of steel, which need to be cooled down to a desired temperature using water injectors before moving on to the next stage. However, there is always a limit on how much water each injector can supply. This limit can be represented by constraints on the control, where the injector is represented by the control.

In this Chapter two problems will be introduced: linear - in section 2.1 and semi-linear - in section 2.2. Then the semi-linear elliptic boundary value problem will be analysed, since the linear case is a particular type of the semi-linear. We will start by analysing existence and uniqueness of solution to the given PDE 2.10b in section 2.3. In the next section 2.4, the existance of an optimal control will be shown. Finally, the optimality conditions will be derived in the last section 2.5.

2.1 Convex problem

Let us consider an area $\Omega \subset \mathbb{R}^3$, that has to be cooled or heated, where $\Omega = [0,1]^2$. The area is governed by a second-order linear elliptic PDE and can be controlled using the heating sources in the domain Ω. The aim of this problem is to choose the control $u(x)$, so that the temperature $y(x)$ is the closest to a desired temperature distribution $y_\Omega(x)$ in Ω, where x represents coordinates x_1 and x_2. The *homogeneous Dirichlet boundary condition* is given on Ω.

This can be modelled in a following way:

$$\min J(y,u) := \frac{1}{2}\int_\Omega |y(x) - y_\Omega(x)|^2 \,\mathrm{d}\Omega + \frac{\lambda}{2}\int_\Omega |u(x)|^2 \,\mathrm{d}\Omega, \tag{2.1a}$$

subject to the *state equation*

$$\begin{cases} -\Delta y(x) = u(x) & \text{in } \Omega \\ \quad y(x) = 0 & \text{on } \Gamma, \end{cases} \tag{2.1b}$$

with the *pointwise control constraints* to the control u

$$u_a(x) \le u(x) \le u_b(x) \qquad u(x) \in U_{ad}. \tag{2.1c}$$

However, in this particular case of the general problem, we investigate a special ansatz 2.1e of the control function, where the control function $u(x)$ is of the form

$$u(x) = \sum_{i=1}^{k} u_i e_i(x), \tag{2.1d}$$

with real values control parameter u_i and finitely many given distribution functions $e_i : \Omega \to \mathbb{R}$ (Figure 2.1). Moreover, we have the *control constraints* to the control u_i

$$u_{\min} \le u_i \le u_{\max} \qquad u_i \in U_{ad,k}, \tag{2.1e}$$

where $u_{\min} < u_{\max}$ are given real numbers. Here U_{ad} is a set of *admissible controls*

$$U_{ad} = \{u \in L^\infty(\Omega) : u_a(x) \le u(x) \le u_b(x) \text{ for a.e. } x \in \Omega\} \tag{2.2}$$

By 2.1e, a special set $U_{ad,k}$ is given, but this set of controls does not have the form of 2.1c,

$$U_{ad,k} = \{u \in L^2(\Omega) : u = \sum_{i=1}^{k} u_i e_i,\ u_{min} \le u_i \le u_{max},\ i = 1,\dots,k,\ u_i \in R\}. \tag{2.3}$$

We take $e_i(x)$ as a basis function

$$e_i(x) = \begin{cases} 1 & , x \in E_i \text{ for } i = 1,\dots,k \\ 0 & ,\text{else} \end{cases} \tag{2.4}$$

where $E_i \subset \Omega$ for $i = 1,\dots,k$ is the set of all (x_1, x_2) with $a_i < x_1 < b_i$ and $c_i < x_2 < d_i$ (See Figure 2.1).

In the given problem the *desired state* $y_\Omega \in L^2(\Omega)$, $u, u_a, u_b \in L^2(\Omega)$ and $u_i \in L^2(\Omega)$ for

$i = 1, \ldots, k$ (for L^2 space see Definition 2.1). $\lambda \geq 0$ represents *regularization parameter*. Depending on it, the optimal solution to the desired one has to change. Regularization parameter needs to be small, between 10^{-3} and 10^{-7}, as the *cost functional* J needs to be minimized. The factor $\frac{1}{2}$, placed in front of the integrals, has no effect on a solution and used only to cancel out factor 2 appearing after differentiation. Δ denotes the Laplacian operator [4]: $\Delta y = \frac{\partial^2 y}{\partial x_1^2} + \frac{\partial^2 y}{\partial x_2^2}$. Also, as explained in a previous section, the pointwise control constraints arise naturally depending on a problem, for example, heating or cooling capacities [5].

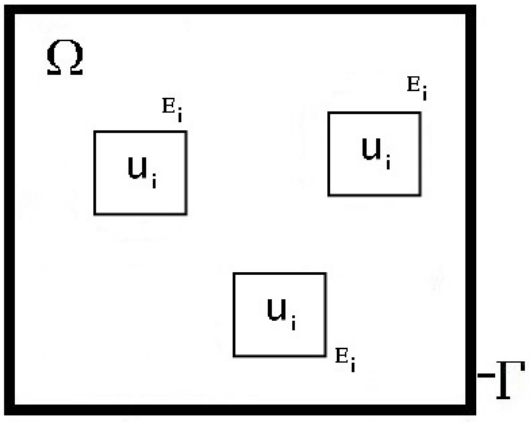


FIGURE 2.1: Distributed control problem with u_i - control parameter and E_i - set of indexes of the nodes of distribution i

The aim is to find optimal control u together with associated state y.

This problem is called *linear-quadratic elliptic distributed control problem*, because: J is a quadratic functional, the state y is governed by a linear elliptic PDE and the control acts on a domain Ω [5].

Lets define the space L^p as follows:

Definition 2.1. *(See Chapter 2.2 in [5])The linear space of all (equivalence classes of) Lebesgue measurable functions $y : \Omega \to \mathbb{R}$ that satisfy*

$$\int_\Omega |y(x)|^p \, \mathrm{d}x < \infty \tag{2.5}$$

is denoted by $L^P(\Omega)$ for $1 \leq p < \infty$. Endowed with the corresponding norm

$$\|y\|_{L^P(\Omega)} = \left(\int_\Omega |y(x)|^p \, \mathrm{d}x \right)^{\frac{1}{p}}. \tag{2.6}$$

Here $L^2(\Omega)$ space is a real Hilbert space , where

$$\|y\|_{L^2(\Omega)} = \sqrt{(y, y)_{L^2(\Omega)}} \tag{2.7}$$

is a norm and

$$(y, y)_{L^2(\Omega)} = \int_\Omega y(x)y(x)\,\mathrm{d}x \tag{2.8}$$

is a scalar product.

Definition 2.2. *(See Chapter 2.2 in [5]) For $L^\infty(\Omega)$ the Banach space of all (equivalence classes of) Lebesgue measurable and bounded functions is denoted by the norm:*

$$\|y\|_{L^\infty(\Omega)} = \operatorname*{ess\,sup}_{x\in\Omega} |y(x)| = \inf_{|F|=0} \left(\sup_{x\in\Omega\setminus F} |y(x)| \right), \tag{2.9}$$

where "ess sup" means essential supremum.

2.2 Non-convex problem

However, not all the problems are linear. Sometimes semi-linear or quasilinear equations are required to model a more realistic simulation [5]. Lets introduce a second problem: *a second-order semi-linear elliptic distributed control problem.* It is modelled in the same way as 2.1, except a non-linear function $d(x, y)$ is added in Ω to the state equation:

$$\min J(y, u) := \frac{1}{2}\int_\Omega |y(x) - y_\Omega(x)|^2\,\mathrm{d}\Omega + \frac{\lambda}{2}\int_\Omega |u(x)|^2\,\mathrm{d}\Omega, \tag{2.10a}$$

subject to the *state equation*

$$\begin{cases} -\Delta y(x) + d(x, y) = u(x) & \text{in } \Omega \\ y(x) = 0 & \text{on } \Gamma, \end{cases} \tag{2.10b}$$

with the *pointwise control constraints* to the control u

$$u_a(x) \le u(x) \le u_b(x) \qquad u(x) \in U_{ad}. \tag{2.10c}$$

Here, as in linear case, we investigate a special ansatz 2.10e of the control function, where the control function $u(x)$ is of the form

$$u(x) = \sum_{i=1}^{k} u_i e_i(x), \tag{2.10d}$$

with real values control parameter u_i and finitely many given distribution functions $e_i : \Omega \to \mathbb{R}$ (Figure 2.1). Moreover, we have the *control constraints* to the control u_i

$$u_{\min} \le u_i \le u_{\max} \qquad u_i \in U_{ad,k}, \tag{2.10e}$$

where $u_{\min} < u_{\max}$ are given real numbers. The sets of *admissible controls* are the same as for the convex problem in Chapter 2.1:

$$U_{ad} = \{u \in L^\infty(\Omega) : u_a(x) \leq u(x) \leq u_b(x) \text{ for a.e. } x \in \Omega\} \tag{2.11}$$

By 2.10e, a special set $U_{ad,k}$ is given, but this set of controls does not have the form of 2.10c,

$$U_{ad,k} = \{u \in L^2(\Omega) : u = \sum_{i=1}^{k} u_i e_i,\ u_{min} \leq u_i \leq u_{max},\ i = 1, \dots, k,\ u_i \in R\}. \tag{2.12}$$

We also take $e_i(x)$ as a basis function

$$e_i(x) = \begin{cases} 1 & ,x \in E_i \text{ for } i = 1, \dots, k \\ 0 & ,\text{else} \end{cases} \tag{2.13}$$

where $E_i \subset \Omega$ for $i = 1, \dots, k$ is the set of all (x_1, x_2) with $a_i < x < b_i$ and $c_i < y < d_i$ (See Figure 2.1).

Here, the same as in a linear case, *desired state* $y_\Omega \in L^2(\Omega)$, $u, u_a, u_b \in L^2(\Omega)$ and $u_i \in L^2(\Omega)$ for $i = 1, \dots, k$ (for L^2 space see Definition 2.1).

Since the linear case is a particular case of the semi-linear, only the theory for semi-linear equation will be used to show existence of a unique solution and optimal control as well as to derive the optimality conditions.

2.3 Existence and uniqueness of solution

The cost functional 2.10a needs to be minimized in order to satisfy the state equation 2.10b and constraints 2.10c on a control.

Equation $-\Delta y + d(x, y) = u$ can not have a *classical solution* $y \in C^2(\Omega) \cap C(\bar{\Omega})$ as control $u \in L^2(\Omega)$. Therefore, we will look for the *weak formulation* of the PDE and the *weak solution* y in the space $H_0^1(\Omega)$ instead.

The following theory works for all convex, closed sets $U_{ad} \subset L^2(\Omega)$, in particular : by 2.1c and 2.10c or by special ansatz 2.1e and 2.10e.

First of all, H^1 space and some necessary assumptions need to be defined:

Definition 2.3. *(See Chapter 2.2.3 in [5])* $H^1(\Omega)$ *is a Hilbert space, defined as*

$$H^1(\Omega) = \{y \in L^2(\Omega) : \mathrm{D}_i y \in L^2(\Omega), i = 1, \dots, N\}, \tag{2.14}$$

with the corresponding norm

$$\|y\|_{H^1(\Omega)} = \left(\int_\Omega (y^2 + |\nabla y|^2) \, \mathrm{d}x \right)^{\frac{1}{2}}, \tag{2.15}$$

where

$$|\nabla y|^2 = (\mathrm{D}_1 y)^2 + \cdots + (\mathrm{D}_N y)^2, \tag{2.16}$$

and the scalar product

$$(u, v)_{H^1(\Omega)} = \int_\Omega uv \, \mathrm{d}x + \int_\Omega \nabla u \cdot \nabla v \, \mathrm{d}x. \tag{2.17}$$

Here $H_0^1(\Omega)$ *is defined as*

$$H_0^1(\Omega) = \{y \in H^1(\Omega) : y|_\Gamma = 0\}, \tag{2.18}$$

with the corresponding norm

$$\|y\|^2_{H_0^1(\Omega)} = \int_\Omega |\nabla y|^2 \, \mathrm{d}x, \tag{2.19}$$

where "$y|_\Gamma = 0$*" means that* y *is* 0 *on a boundary* Γ*.*

Assumption 2.4. *(See Assumption 4.2 in [5]) Let the following assumptions be fulfilled:*

(i) $\Omega \subset \mathbb{R}^N$*,* $N \geq 2$ *is a bounded Lipschitz domain.*

(ii) For all $y \in \mathbb{R}$ *the non-linear function* $d = d(x, y) : \Omega \times \mathbb{R} \to \mathbb{R}$ *is bounded, measurable and twice continuous differentiable with respect to* $x \in \Omega$*, for every fixed* $y \in \mathbb{R}$*. Furthermore, it is locally Lipschitz continues and monotone increasing in* y *for almost all* $x \in \Omega$*.*

Assumption 2.5. *(See Assumption 4.3 in [5])*

(i) Let $d(x, 0) = 0$ *for almost all* $x \in \Omega$ *and* d *be globally bounded, i.e. there is a positive constant* M*, such that for all* $y \in \mathbb{R}$

$$|d(x, y)| \leq M \quad \text{for a.e } x \in \Omega$$

Then follows the definition of a *weak derivative*:

Definition 2.6. *(See p.27 in [5]) Let* $y \in L^1_{loc}(\Omega)$*, which denotes the set of all locally integrable functions in* Ω*, be given together with some multi-index* α*. If a function*

$w \in L^1_{loc}(\Omega)$ satisfies

$$\int_\Omega y(x) D^\alpha v(x)\,\mathrm{d}\Omega = (-1)^{|\alpha|} \int_\Omega w(x) v(x)\,\mathrm{d}\Omega \qquad \text{for all } v \in C_0^\infty(\Omega)$$

then w is called the weak derivative of y (associated with α).

In order to derive the weak formulation, both sides of the semi-linear elliptic boundary value problem 2.10b are multiplied by a test function $v \in H_0^1(\Omega)$ and integrated over Ω:

$$-\int_\Omega \Delta y v\,\mathrm{d}\Omega + \int_\Omega d(x,y) v\,\mathrm{d}\Omega = \int_\Omega uv\,\mathrm{d}\Omega. \tag{2.20}$$

After using integration by parts, *Gauss' divergence theorem* [4] and consideration of the boundary conditions we get the following:

$$-\int_\Gamma \frac{\partial y}{\partial n} v\,\mathrm{d}\Gamma + \int_\Omega \nabla y \cdot \nabla v\,\mathrm{d}\Omega + \int_\Omega d(x,y) v\,\mathrm{d}\Omega = \int_\Omega uv\,\mathrm{d}\Omega, \tag{2.21}$$

where n is the outward unit normal to Γ. Since v vanishes on Γ, therefore the weak formulation is:

$$\int_\Omega \nabla y \cdot \nabla v\,\mathrm{d}\Omega + \int_\Omega d(x,y) v\,\mathrm{d}\Omega = \int_\Omega uv\,\mathrm{d}\Omega, \quad \forall v \in H_0^1(\Omega). \tag{2.22}$$

Definition 2.7. *[5] Let the Assumptions 2.4 and 2.5 hold. Then a state equation 2.10b has a weak solution $y \in H_0^1(\Omega)$, if it satisfies the weak formulation 2.22 for all $v \in H_0^1(\Omega)$.*

Theorem 2.8. *(See Theorem 4.4 in [5]) Suppose that Assumption 2.4 and 2.5 hold. Then for all $u \in L^2(\Omega)$, the elliptic boundary value problem 2.10b has a unique weak solution $y \in H_0^1(\Omega)$.*

For the proof, we refer to the chapter 4.2.2 in [5].

However, the Theorem 2.8 does not work for $d(x,y) = y^3$, or y^5, as Assumption 2.5, which states that $d(x,y)$ is bounded cannot be satisfied. Therefore, let as use only Assumption 2.4 and state that in this case the PDE 2.10b has a weak solution $y \in H_0^1(\Omega) \cap L^\infty(\Omega)$ if it satisfies the weak formulation 2.22.

Theorem 2.9. *(See Theorem 4.7 in [5]) Suppose that Assumption 2.4 hold and $d(x,y) = 0$ for almost all $x \in \Omega$. Then for all $u \in L^r(\Omega)$, where $r > \frac{N}{2}$ the semi-linear elliptic boundary value problem 2.10b has a unique weak solution $y \in H_0^1(\Omega) \cap L^\infty(\Omega)$. The solution y is also continuous on $\bar{\Omega}$.*

Again, for the proof, we refer to the chapter 4.2.3 in [5].

Remark 2.10. *Theorem 2.9 works also for u with special ansatz.*

Based on the previous theorem, we introduce linear and continuous *control-to-state operator* which maps the control function u to the state y:

$$G : L^r(\Omega) \to H_0^1(\Omega) \cap C(\bar{\Omega}) \qquad u \mapsto y(u). \tag{2.23}$$

We will also introduce embedding operator A, where $A : H_0^1(\Omega) \cap C(\bar{\Omega}) \to L^2(\Omega)$, which assigns to each function $y \in H_0^1(\Omega) \cap C(\bar{\Omega})$ the same function in $L^2(\Omega)$. Here A is also linear and continuous operator. We define new operator S by multiplying G by A:

$$S = GA, \tag{2.24}$$

which maps the control function u to the state y

$$S : L^r(\Omega) \to L^2(\Omega) \qquad u \mapsto y(u). \tag{2.25}$$

Therefore, we consider the reduced cost functional f of 2.10a, where $y = Su$

$$J(y, u) = J(Su, u) = f(u) \tag{2.26}$$

$$\min_{u \in U_{ad}} f(u) = \frac{1}{2}\|Su - y_\Omega\|^2_{L^2(\Omega)} + \frac{\lambda}{2}\|u\|^2_{L^2(\Omega)} \tag{2.27}$$

As for all bounded u, we have a unique weak solution y. Hence the same applies for $\vec{u}$ in 2.1d, constructed as the finite sum of the e_i.

2.4 Existence of optimal control

As we have found out in the previous section, semi-linear elliptic PDE 2.10b has a unique solution $y \in H_0^1(\Omega) \cap L^\infty(\Omega)$ for $u \in L^r(\Omega)$, where $r > \frac{N}{2}$. Now we need to check if an optimal control for the problem 2.10 exists.

Let us start with the definition of existence of optimal control, followed by some properties of operator G.

Definition 2.11. *(See [5]) A state $\bar{y} = G\bar{u}$ is called the optimal state associated with $\bar{u}$ and control $\bar{u} \in U_{ad}$ is called an optimal control if $f(\bar{u}) \leq f(u)$ for all $u \in U_{ad}$.*

Here U_{ad}, $U_{ad,k}$ and $\vec{U}_{ad,k}$ are the sets of *admissible controls* and are given by

$$U_{ad} = \{u \in L^\infty(\Omega) : u_a(x) \leq u(x) \leq u_b(x) \text{ for a.e. } x \in \Omega\}, \tag{2.28}$$

$$U_{ad,k} = \{u \in L^2(\Omega) : u = \sum_{i=1}^{k} u_i e_i,\ u_{min} \leq u_i \leq u_{max},\ i = 1, \ldots, k,\ u_i \in R\}, \tag{2.29}$$

$$\vec{U}_{ad,k} = \{\vec{u} : u_{min} \leq u_i \leq u_{max},\ i = 1, \ldots, k\}. \tag{2.30}$$

We will be using Assumption 2.4 to define properties of G in the following theorem:

Theorem 2.12. *(See Theorem 4.16 in [5]) Let the Assumption 2.4 be satisfied. For $r > \frac{N}{2}$ the mapping G is Lipschitz continuous from $L^r(\Omega)$ into $H_0^1(\Omega) \cap C(\bar{\Omega})$ with $L > 0$ such that*

$$\|y_1 - y_2\|_{H_0^1(\Omega)} + \|y_1 - y_2\|_{C(\bar{\Omega})} \leq L\|u_1 - u_2\|_{L^r(\Omega)} \tag{2.31}$$

for every $u_i \in L^r(\Omega)$ and $y_i = G(u_i)$, $i = 1,2$.

For the proof, we refer to the chapter 4.5.1 in [5].

Theorem 2.13. *(See Theorem 4.17 in [5]) Let the Assumption 2.4 be satisfied. Then for any $r > \frac{N}{2}$ the control-to-state operator G is Frechet differentiable from $L^r(\Omega)$ into $H_0^1(\Omega) \cap C(\bar{\Omega})$. Its directional derivative at $\bar{u} \in L^r(\Omega)$ in the direction u is presented by*

$$G'(\bar{u})u = y, \tag{2.32}$$

where y is the weak solution to the boundary value problem linearised at $\bar{y} = G(\bar{u})$:

$$\begin{cases} -\Delta y + d_y(x, \bar{y})y = u & \text{in } \Omega \\ y = 0 & \text{on } \Gamma \end{cases} \tag{2.33}$$

Again, for the proof, we refer to the chapter 4.5.1 in [5].

Using the theorems above and the assumption we get the following existance theorem for the general case of our optimal control of semi-linear elliptic PDE 2.10.

Theorem 2.14. *(See Theorem 4.15 in [5]) Under Assumption 2.4, problem 2.10 has at least one optimal control $\bar{u}$ with associated optimal state $\bar{y} = y(\bar{u}) \in H_0^1(\Omega) \cap C(\bar{\Omega})$.*

However, lets consider the particular case, where the control function $u(x)$ is expressed as 2.10b

$$\begin{cases} -\Delta y(x) + d(x,y) = \sum_{i=1}^{k} u_i e_i(x) & \text{in } \Omega \\ y(x) = 0 & \text{on } \Gamma. \end{cases} \tag{2.34}$$

Therefore, we need to introduce another linear operator T, where

$$T : R^k \to L^r(\Omega) \qquad \vec{u} \mapsto u(x) = \sum_{i=1}^{k} u_i e_i(x). \tag{2.35}$$

Then we define a new linear operator $\Lambda = AGT$, where

$$G : L^r(\Omega) \to H_0^1(\Omega) \cap C(\bar{\Omega}) \qquad u \mapsto y(u) \tag{2.36}$$

$$A : H_0^1(\Omega) \cap C(\bar{\Omega}) \to L^2(\Omega) \qquad y(u) \mapsto y(u) \tag{2.37}$$

Here,

$$\Lambda : R^k \to L^2(\Omega) \qquad \vec{u} \mapsto y(u), \tag{2.38}$$

with its adjoint operator

$$\Lambda^* : L^2(\Omega) \to R^k \qquad y(u) \mapsto \vec{u}. \tag{2.39}$$

Therefore, we consider the reduced cost functional f of 2.10a, where $y = \Lambda\vec{u}$

$$\min_{u \in \vec{U}_{ad,k}} f(\vec{u}) = \frac{1}{2}\|\Lambda\vec{u} - y_\Omega\|^2_{L^2(\Omega)} + \frac{\lambda}{2}\|u\|^2_{L^2(\Omega)} \tag{2.40}$$

Using Definition 2.11, vector $\vec{\bar{u}} \in \vec{U}_{ad,k}$ is called an optimal control vector if $f(\vec{\bar{u}}) \leq f(\vec{u})$ for all $\vec{u}$.

Proof: The set of admissible controls $\vec{U}_{ad,k}$ is bounded and closed, hence compact. The mapping from $\vec{u}$ to the objective functional value is continuous. Therefore, by using *Weierstrass theorem* we get that $\exists\vec{\bar{u}} \in \vec{U}_{ad,k}$ that

$$f(\vec{\bar{u}}) = \min_{\vec{u} \in \vec{U}_{ad,k}} f(\vec{u}) \tag{2.41}$$

2.5 Necessary optimality conditions

As it was proved in the previous section, our given semi-linear elliptic PDE 2.10 has at least one optimal control $\bar{u}$ and the associated optimal state $\bar{y}$. Now we need to

check what conditions these two vectors satisfy. We will do so by finding the first order optimality conditions for the given PDE. These need to be satisfied by both vectors $\bar{u}$ and $\bar{y}$.

The necessary optimality conditions can be derived using the formal *Lagrange method*, where $\mathcal{L}$ represents the *Lagrangian function* with its parameter p. Using system of equations 2.10, the function is given as follows:

$$\mathcal{L}(y,u,p) = J(y,u) - \int_\Omega (-\Delta y + d(x,y) - u)p \,\mathrm{d}\Omega - \int_\Gamma yp \,\mathrm{d}\Gamma \tag{2.42}$$

From PDE 2.10b, $y = 0$ on a boundary, thus the last term disappears. Furthermore, we apply integration by parts and *Gauss divergence theorem* for Δy. Therefore, we get the following:

$$\mathcal{L}(y,u,p) = J(y,u) + \int_\Omega (y\Delta p - d(x,y)p + up) \,\mathrm{d}\Omega - \int_\Gamma y\frac{\partial p}{\partial n} \,\mathrm{d}\Gamma \tag{2.43}$$

After computing the derivatives of $\mathcal{L}$, we find that the second and third conditions in the optimality system are as follows:

$$\nabla_y \mathcal{L}(\bar{y},\bar{u},p)y = 0 \tag{2.44}$$

$$\nabla_u \mathcal{L}(\bar{y},\bar{u},p)(u-\bar{u}) \geq 0 \quad \forall u \in U_{ad} \tag{2.45}$$

We need to solve inequalities 2.44 and 2.45 to get the necessary optimality conditions:

$$\begin{aligned}\mathcal{L}_y(\bar{y},\bar{u},p)y &= \int_\Omega (\bar{y} - y_\Omega)y \,\mathrm{d}\Omega + \int_\Omega (y\Delta p - d_y(x,\bar{y})py) \,\mathrm{d}\Omega - \int_\Gamma y\frac{\partial p}{\partial n} \,\mathrm{d}\Gamma \\ &= \int_\Omega (\bar{y} - y_\Omega + \Delta p - d_y(x,\bar{y})p)y \,\mathrm{d}\Omega - \int_\Gamma y\frac{\partial p}{\partial n} \,\mathrm{d}\Gamma\end{aligned}$$

$$\begin{aligned}\mathcal{L}_u(\bar{y},\bar{u},p)(u-\bar{u}) &= \int_\Omega \lambda\bar{u}(u-\bar{u}) \,\mathrm{d}\Omega + \int_\Omega p(u-\bar{u}) \,\mathrm{d}\Omega \\ &= \int_\Omega (p + \lambda\bar{u})(u-\bar{u}) \,\mathrm{d}\Omega \geq 0\end{aligned}$$

Therefore:

$$\mathcal{L}_y(\bar{y},\bar{u},p)y = \int_\Omega (\bar{y} - y_\Omega + \Delta p - d_y(x,\bar{y})p)y \,\mathrm{d}\Omega - \int_\Gamma y\frac{\partial p}{\partial n} \,\mathrm{d}\Gamma \tag{2.46}$$

$$\mathcal{L}_u(\bar{y},\bar{u},p)(u-\bar{u}) = \int_\Omega (p + \lambda\bar{u})(u-\bar{u}) \,\mathrm{d}\Omega \geq 0 \tag{2.47}$$

Equations 2.46 and 2.47 give the necessary optimality conditions:

State equation:

$$\begin{cases} -\Delta y + d(x,y) = u & \text{in } \Omega \\ y = 0 & \text{on } \Gamma \end{cases} \tag{2.48a}$$

Equation 2.46 is a weak formulation for the following *adjoint equation*, which is the linearized state equation:

$$\begin{cases} -\Delta p + d_y(x,\bar{y})p = y - y_\Omega & \text{in } \Omega \\ p = 0 & \text{on } \Gamma \end{cases} \tag{2.48b}$$

Equation 2.47 is the *variational inequality* for every optimal control $\bar{u}$ with associated adjoint state $p \in H_0^1(\Omega) \cap C(\bar{\Omega})$ for the general case:

$$\int_\Omega (p + \lambda\bar{u})(u - \bar{u})\,\mathrm{d}\Omega \geq 0 \quad \forall u \in U_{ad} \tag{2.48c}$$

Variational inequality 2.48c can be reformulated as a minimization problem [5]:

$$\int_\Omega (p + \lambda\bar{u})\bar{u}\,\mathrm{d}\Omega \leq \int_\Omega (p + \lambda\bar{u})u\,\mathrm{d}\Omega, \tag{2.49}$$

hence

$$\int_\Omega (p + \lambda\bar{u})\bar{u}\,\mathrm{d}\Omega = \min_{u \in U_{ad}} \int_\Omega (p + \lambda\bar{u})u\,\mathrm{d}\Omega \quad \forall u \in U_{ad} \tag{2.50}$$

Theorem 2.15. *(See Chapter 4.6 in [5])Let the Assumption 2.4 hold and let $\bar{u}$ be a locally optimal control for the problem 2.10 and p be the associated adjoint state. Then the minimum of the 2.50 problem for a.e.$x \in \Omega$*

$$\min_{u \in U_{ad}} \int_\Omega (p(x) + \lambda\bar{u}(x))u\,\mathrm{d}\Omega \quad \forall u \in U_{ad} \tag{2.51}$$

is attained at $u = \bar{u}(x)$.

For $\lambda > 0$, the projection formula is as follows:

$$\bar{u}(x) = \mathbb{P}_{[u_a(x), u_b(x)]}\Big\{ -\frac{1}{\lambda}p(x) \Big\} \quad \text{for a. e. } x \in \Omega \tag{2.52}$$

Therefore all the necessary optimality conditions have been derived in 2.48.

Now, let us find the variational inequality for the particular control function 2.10d by inserting it into 2.48c

$$\begin{aligned}
&\int_\Omega (p+\lambda \sum_{j=1}^{k} \bar{u}_j e_j)(\sum_{i=1}^{k}(u_i-\bar{u}_i)e_i)\,\mathrm{d}\Omega = \\
&\quad = \int_\Omega \sum_{i=1}^{k} p(u_i-\bar{u}_i)e_i)\,\mathrm{d}\Omega + \lambda \int_\Omega \sum_{i,j=1}^{k} \bar{u}_j e_j (u_i-\bar{u}_i)e_i\,\mathrm{d}\Omega \\
&\quad = \sum_{i=1}^{k}(u_i-\bar{u}_i)\Big(\int_\Omega p(x)e_i(x)\,\mathrm{d}\Omega\Big) + \lambda \sum_{i,j=1}^{k} \bar{u}_j(u_i-\bar{u}_i)\int_\Omega e_i e_j\,\mathrm{d}\Omega \\
&\quad = (\vec{u}-\vec{\bar{u}})\cdot \begin{bmatrix} \int_\Omega p e_1\,\mathrm{d}\Omega \\ \vdots \\ \int_\Omega p e_k\,\mathrm{d}\Omega \end{bmatrix} + \lambda \vec{\bar{u}}\cdot D(\vec{u}-\vec{\bar{u}}) \geq 0 \qquad \forall \vec{u} \in \vec{U}_{ad,k}
\end{aligned} \tag{2.53}$$

where

$$\vec{p} = \begin{bmatrix} \int_\Omega p e_1\,\mathrm{d}\Omega \\ \vdots \\ \int_\Omega p e_k\,\mathrm{d}\Omega \end{bmatrix} \quad \text{and} \quad D_{ij} = \int_\Omega e_i e_j\,\mathrm{d}\Omega. \tag{2.54}$$

Then the *variational inequality* for optimal vector $\vec{\bar{u}}$

$$(\vec{p}+\lambda D^T \vec{\bar{u}}, \vec{u}-\vec{\bar{u}})_{\mathbb{R}^k} \geq 0 \qquad \forall \vec{u} \in \vec{U}_{ad,k} \tag{2.55}$$

Then for $\lambda > 0$, the associated projection formula becomes:

$$\bar{u}_i = \mathbb{P}_{[u_{min},u_{max}]}\Big\{-\frac{1}{\lambda} D_{ii}\bar{p}_i\Big\} \quad i = 1,\ldots,k \tag{2.56}$$

Chapter 3

Finite Volume Method

In Chapter 2 optimal control problems governed by second-order linear and semi-linear elliptic partial differential equations were introduced. Semi-linear problem was analysed: existence of the unique solution and optimal control, as well as necessary optimality conditions was proven. However, these PDEs are still in the infinite dimension. Therefore, the next step would be to discretize both of the PDEs 2.1b and 2.10b, to obtain a finite dimensional problem. Let us not forget that the discretization must be very fine, in order to find an acceptable approximation. In Chapter 3 and the following Chapter 4 we will introduce and apply two discretization methods. We will start by *Finite Volume Method* (FVM), which is the most popular amongst engineers. It solves the PDE after integrating it over the control volume. This is a standard advanced numerical method. More information and examples of this method can be found in [4, 15].

In section 3.1 we will start with an introduction to the FVM. Then in section 3.2 linear elliptic PDE 2.1b will be discretized, followed by a semi-linear PDE 2.10b in section 3.4. However, the semi-linear part causes an additional problem, therefore, the *Newton's method* will be introduced and applied to the equation in section 3.3 in order to linearise it first.

3.1 Mesh of the unit square

We will consider a rectangular domain $\Omega = [0, 1]^2$ governed by a PDE with homogeneous *Dirichlet* boundary condition, which means $y = 0$ on a boundary. A square mesh is applied to the domain and the unknown values of y are placed on the nodes. In Figure 3.1 y is represented by red and black points: red - internal nodes, black - boundary nodes. In the same figure m represents the number of points in either direction of the mesh.

We have chosen a horizontal numbering style, where values of the vector y are placed sequentially on the nodes (Figure 3.2).

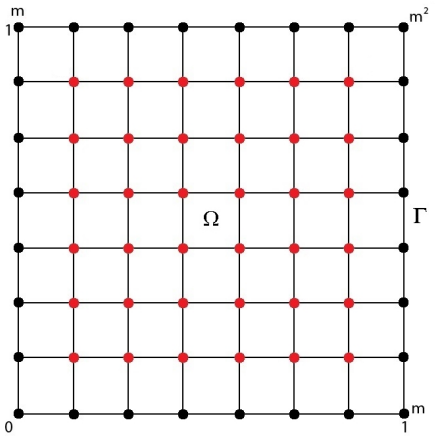


FIGURE 3.1: Mesh of the domain Ω. Red nodes - internal, black nodes - boundary

31	32	33	34	35	36	
25	26	27	28	29	30	
19	20	21	22	23	24	
13	14	15	16	17	18	
7	8	9	10	11	12	
1	2	3	4	5	6	

FIGURE 3.2: Horizontal numbering style of the domain Ω

Let us consider one node of the vector y and call it j. We use the 4 surrounding neighbouring nodes of y and denote them as follows: right - "j+1", left - "j-1", above - "j+m", bellow - "j-m". Then the control volume Ω_j, with edges in the middle of two nodes y, is constructed. It can be seen in the Figure 3.3, where $h = \frac{1}{m-1}$ represents the distance between two nodes. For the sake of simplicity, h is chosen to be the same in x_1 and x_2 direction.

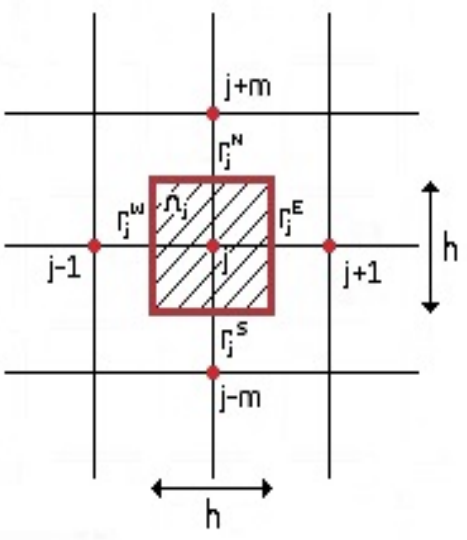


FIGURE 3.3: Control volume Ω_j

3.2 Discretization of the linear boundary value problem

Lets consider the linear second-order elliptic partial differential equation 2.1b described in Chapter 2.1:

$$\begin{cases} -\Delta y(x) = u(x) & x \in \Omega \\ \quad y(x) = 0 & x \in \Gamma, \end{cases} \tag{3.1}$$

where the control function $u(x)$ is of the form

$$u(x) = \sum_{i=1}^{k} u_i e_i(x), \tag{3.2}$$

with control parameter u_i, which has real variables, and finitely many given distribution functions $e_i : \Omega \to \mathbb{R}$

We take $e_i(x)$ as described in Chapter 2.1

$$e_i(x) = \begin{cases} 1 & , x \in E_i \text{ for } i = 1, \ldots, k \\ 0 & ,\text{else} \end{cases} \tag{3.3}$$

where $E_i \subset \Omega$ for $i = 1, \ldots, k$ is the set of all (x_1, x_2) with $a_i < x < b_i$ and $c_i < y < d_i$ (See Figure 2.1).

We integrate the equation 3.1 over the control volume Ω_j

$$-\int_{\Omega_j} \Delta y \, \mathrm{d}\Omega = \int_{\Omega_j} \sum_{i=1}^{k} u_i e_i \, \mathrm{d}\Omega \tag{3.4}$$

Apply Gauss' divergence theorem

$$-\int_{\Gamma_j} \frac{\partial y}{\partial n} d\Gamma = \int_{\Omega_j} \sum_{i=1}^{k} u_i e_i \, \mathrm{d}\Omega \tag{3.5}$$

Using central differences for $\frac{\partial y}{\partial n}$ and middle point integration rule for the right hand side, we get:

$$\oint_{\Gamma_j} = \int_{\Gamma_j^S} + \int_{\Gamma_j^E} + \int_{\Gamma_j^N} + \int_{\Gamma_j^W} \tag{3.6}$$

where,

$$-\int_{\Gamma_j^S} \frac{\partial y}{\partial n} d\Gamma \approx \frac{y_j - y_{j-m}}{h} h = y_j - y_{j-m} \tag{3.7}$$

$$-\int_{\Gamma_j^E} \frac{\partial y}{\partial n} d\Gamma \approx \frac{y_j - y_{j+1}}{h} h = y_j - y_{j+1} \tag{3.8}$$

$$-\int_{\Gamma_j^N} \frac{\partial y}{\partial n} d\Gamma \approx \frac{y_j - y_{j+m}}{h} h = y_j - y_{j+m} \tag{3.9}$$

$$-\int_{\Gamma_j^W} \frac{\partial y}{\partial n} d\Gamma \approx \frac{y_j - y_{j-1}}{h} h = y_j - y_{j-1} \tag{3.10}$$

$$\int_{\Omega_j} \sum_{i=1}^{k} u_i e_i \, \mathrm{d}\Omega = \begin{cases} u_i h^2 & \text{, for all nodes} \in E_i \text{ for } i = 1, \ldots, k \\ 0 & \text{,else} \end{cases} \tag{3.11}$$

Remark 3.1. $y(x_j) = y_j$ *everywhere.*

Adding equations 3.7 to 3.10 and 3.11 together gives the *discretization equation*:

$$-y_{j-1} - y_{j+1} + 4y_j - y_{j-m} - y_{j+m} = \begin{cases} u_i h^2 & \text{, for all nodes} \in E_i \text{ for } i = 1, \ldots, k \\ 0 & \text{,else} \end{cases} \tag{3.12}$$

Therefore, we end up in a matrix-vector form

$$A\vec{y} = \vec{u}, \tag{3.13}$$

where $A \in \mathbb{R}^{m \times m}$ is a sparse matrix and $\vec{y}, \vec{u} \in \mathbb{R}^{m \times 1}$.

As it was mentioned before, in order to get an acceptable approximation, we need to use a very fine mesh (m is inversely proportional to h). Hence, mathematical programming software MATLAB will be used for the discretization of the PDE 3.1 applying equation 3.12. The use of MATLAB for PDEs is extensively elaborated in [16, 17].

First of all, let us consider a simple Example 1, where $k = 1$, $u_i = 1$, $E_i = \Omega$ (contains all the node points y) for $i = 1, \ldots, k$ and $m = 30$. Therefore, the control function $u(x) = 1$ for all x in domain Ω:

$$\begin{cases} -\Delta y(x) = u(x) & x \in \Omega \\ y(x) = 0 & x \in \Gamma \end{cases} \tag{3.14}$$

The discretization equation for PDE 3.14 is as follows:

$$-y_{j-1} - y_{j+1} + 4y_j - y_{j-m} - y_{j+m} = 1 \cdot h^2 \tag{3.15}$$

By discretizing this PDE using FVM on MATLAB and visually representing the solution vector y for given settings we get the following results in Figure 3.4.

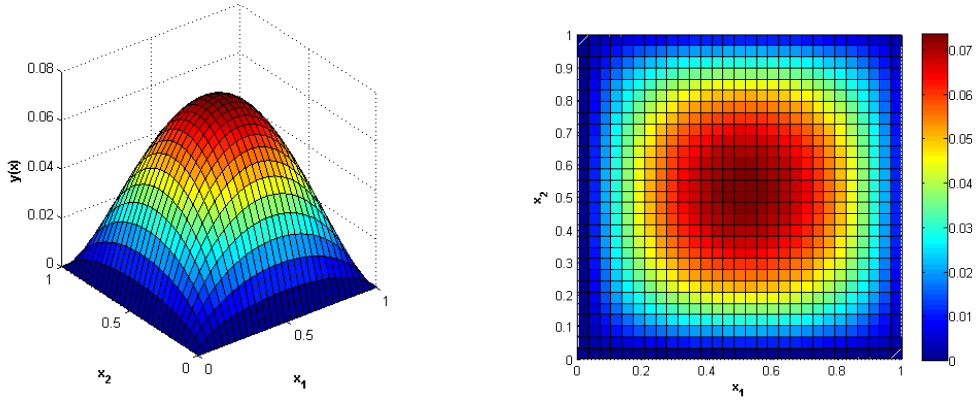


FIGURE 3.4: Example 1 - discretized solution of the linear PDE 3.14 using FVM, for $m = 30$, $k = 1$, $u_i = 1$ and $E_i = \Omega$ for $i = 1, \ldots, k$.

Next, let us consider three more examples for PDE 3.1 with its discretization equation 3.12. We choose $u_i = 100$, to highlight the difference from the semi-linear part,and $m = 72$ (number of internal points in either direction of the mesh) for $i = 1, \ldots, k$:

Example 2: $k = 9$ and $E_i = 4$ - means E_i is a square composed of 4 cells;
Example 3: $k = 9$ and $E_i = 16$ - means E_i is a square composed of 16 cells;
Example 4: $k = 4$ and $E_i = 36$ - means E_i is a square composed of 36 cells;

Again, we discretize this PDE using FVM on MATLAB and visually represent the solution vector y in 3D and 2D. Example 2 is shown in Figure 3.5, where we have a discretized solution of 9 distribution functions e_i. Each area E_i is a square composed of 4 cells for $i = 1, \ldots, k$. Examples 3 and 4 can be seen in Figures 3.6 and 3.7, respectively.

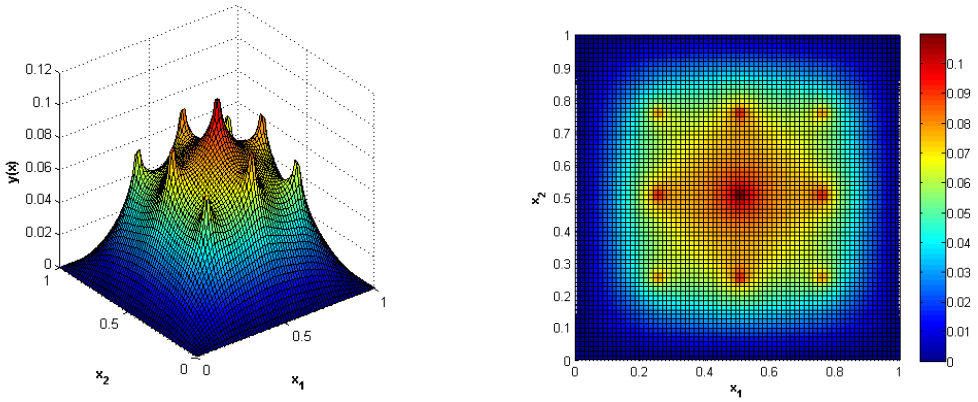


FIGURE 3.5: Example 2 - discretized solution of the linear PDE 3.1 using FVM, for $m = 72$, $k = 9$, $u_i = 100$ and $E_i = 4$ for $i = 1, \ldots, k$.

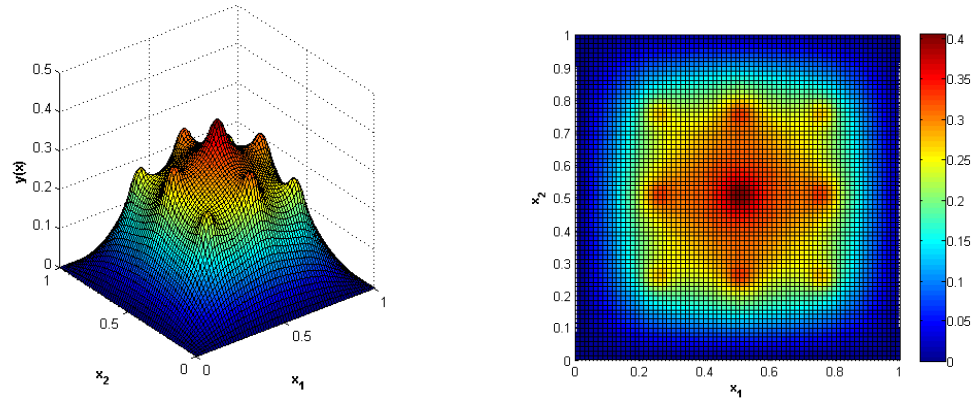


FIGURE 3.6: Example 3 - discretized solution of the linear PDE 3.1 using FVM, for $m = 72$, $k = 9$, $u_i = 100$ and $E_i = 16$ for $i = 1, \ldots, k$.

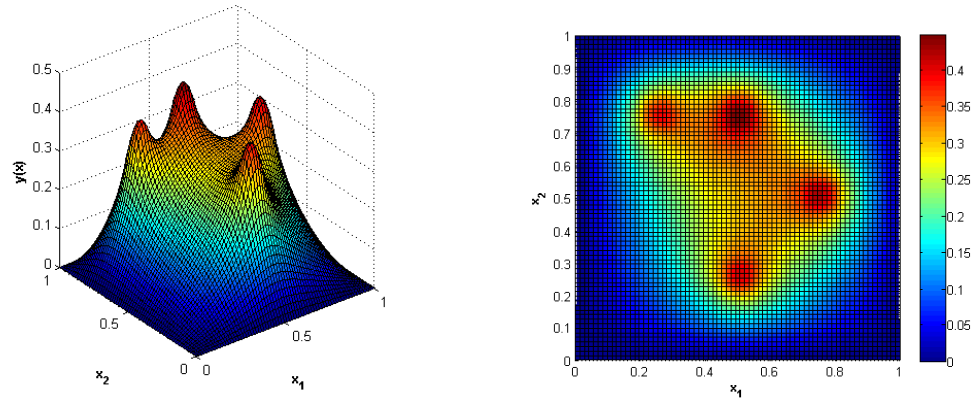


FIGURE 3.7: Example 4 - discretized solution of the linear PDE 3.1 using FVM, for $m = 72$, $k = 4$, $u_i = 100$ and $E_i = 36$ for $i = 1, \ldots, k$.

In fact, we can not discretize semi-linear elliptic PDE 2.10b directly, as it has a semi-linear part. Therefore, in the next section we will learn how to linearize it first.

3.3 Newton's method

There are many different methods which help to deal with semi-linear PDEs. *Picard's* and *Newton's methods* are among the most popular classical iterative methods. In general, *Picard's method* converges linearly, while *Newton's* - quadratically in a neighbourhood of the root, provided that the initial guess was chosen correctly [4]. In this section we will introduce and apply *Newton's method* to our given semi-linear PDE 2.10b, as it converges much faster than *Picard's*.

Lets consider a semi-linear second-order elliptic partial differential equation 2.10b described in Chapter 2.2.

$$\begin{cases} -\Delta y(x) + d(x,y) = u(x) & \text{in } \Omega \\ y(x) = 0 & \text{on } \Gamma \end{cases} \tag{3.16}$$

Suppose we have a current approximation of the solution y^n of the equation 3.16 with the goal that $y^n \to y$ as $n \to \infty$. Every estimate y^n is obtained by solving a linear system of equations.

Let us write equation 3.16 as function $f(y)$:

$$f(y) = -\Delta y + d(y) - u \tag{3.17}$$

Next we calculate the derivative of the function 3.17:

$$f'(y) = d'(y) \tag{3.18}$$

Then we can derive a formula for a better approximation y^{n+1} [4]:

$$f'(y^n)(y^{n+1} - y^n) + f(y^n) = 0 \tag{3.19}$$

After substitution we get the following:

$$-\Delta y^{n+1} + d(y^n) + d'(y^n)(y^{n+1} - y^n) = u \tag{3.20}$$

Putting unknown values to the left hand side and known - to the right, we get the following:

$$-\Delta y^{n+1} + d'(y^n)y^{n+1} = u - d(y^n) + d'(y^n)y^n \tag{3.21}$$

where y^n is the previous known value and y^{n+1} is unknown.

The convergence of the *Newton's method* is sensitive to the good initial guess y^0. There are different ways to choose it. We will find initial guess y^0 by solving PDE 3.16 without the semi-linear part.

3.4 Discretization of the semi-linear boundary value problem

Lets consider the semi-linear second-order elliptic partial differential equation 2.10b described in Chapter 2.2.

$$\begin{cases} -\Delta y(x) + d(x, y) = u(x) & x \in \Omega \\ y(x) = 0 & x \in \Gamma, \end{cases} \tag{3.22}$$

where

$$u(x) = \sum_{i=1}^{k} u_i e_i(x). \tag{3.23}$$

We take $e_i(x)$ as a basis function (the same as in section 3.2).

As the PDE 3.22 is semi-linear, we will be using Newton's method (described in section 3.3) to linearise the equation. First of all we find the initial guess y^0. We can do so by solving the equation without the semi-linear part:

$$\begin{cases} -\Delta y^0(x) = u(x) & x \in \Omega \\ y^0(x) = 0 & x \in \Gamma \end{cases} \tag{3.24}$$

PDE 3.24 is the same as linear PDE 3.1 in Chapter 3.2. After repeating the same steps and substituting control function 3.23 into PDE 3.24, we end up with the following

discretization equation:

$$-y^0_{j-1} - y^0_{j+1} + 4y^0_j - y^0_{j-m} - y^0_{j+m} = \begin{cases} u_i h^2 & \text{, for all nodes } \in E_i \text{ for } i = 1, \dots, k \\ 0 & \text{,else} \end{cases} \tag{3.25}$$

Solution y^0 can be obtained using mathematical software MATLAB. Now, as we found the initial vector y^0, we can use it to start the Newton iteration. After applying Newton's method to the semi-linear PDE 3.22, we get equation (3.21):

$$-\Delta y^{n+1} + d'(y^n)y^{n+1} = \sum_{i=1}^{k} u_i e_i - d(y^n) + d'(y^n)y^n. \tag{3.26}$$

Let us take $d(y) = y^3$. Then we get the following:

$$-\Delta y^{n+1} + 3(y^n)^2 y^{n+1} = \sum_{i=1}^{k} u_i e_i - (y^n)^3 + 3(y^n)^2 y^n \tag{3.27}$$

$$-\Delta y^{n+1} + 3(y^n)^2 y^{n+1} = \sum_{i=1}^{k} u_i e_i + 2(y^n)^3 \tag{3.28}$$

Integrating the equation 3.28 over the control volume Ω_j, as in section 3.2, gives the following result:

$$\int_{\Omega_j} -\Delta y^{n+1} \, \mathrm{d}\Omega = -y^{n+1}_{j-1} - y^{n+1}_{j+1} + 4y^{n+1}_j - y^{n+1}_{j-m} - y^{n+1}_{j+m} \tag{3.29}$$

$$\int_{\Omega_j} 3(y^n)^2 y^{n+1} \, \mathrm{d}\Omega = 3(y^n_j)^2 \int_{\Omega_j} y^{n+1} \, \mathrm{d}\Omega = 3(y^n_j)^2 y^{n+1}_j h^2 \tag{3.30}$$

$$\int_{\Omega_j} 2(y^n)^3 \, \mathrm{d}\Omega = 2(y^n_j)^3 \int_{\Omega_j} \mathrm{d}\Omega = 2(y^n_j)^3 h^2 \tag{3.31}$$

$$\int_{\Omega_j} \sum_{i=1}^{k} u_i e_i \, \mathrm{d}\Omega = \begin{cases} u_i h^2 & \text{, for all nodes } E_i \text{ for } i = 1, \dots, k \\ 0 & \text{,else} \end{cases} \tag{3.32}$$

Inserting all equations from 3.29 to 3.32 in to equation 3.28 gives the *discretization equation*:

$$-y^{n+1}_{j-1} - y^{n+1}_{j+1} + 4y^{n+1}_j - y^{n+1}_{j-m} - y^{n+1}_{j+m} + 3(y^n_j)^2 y^{n+1}_j h^2 = \begin{cases} u_i h^2 + 2(y^n_j)^3 h^2 & \text{in } E_i \\ 2(y^n_j)^3 h^2 & \text{else,} \end{cases} \tag{3.33}$$

where y^n is the previous known value and y^{n+1} is unknown. For comparison, red colour illustrates the differences from the linear discretization equation 3.12.

We will again consider the same three examples (2 - 4) for the semi-linear PDE 3.22 using its discretization equation 3.33: **Ex.2** - $k = 9$, $E_i = 4$; **Ex.3** - $k = 9$, $E_i = 16$; **Ex.4** - $k = 4$, $E_i = 36$. We choose $u_i = 100$ and $m = 72$ for $i = 1, \ldots, k$ and discretize this PDE using FVM on MATLAB. After three Newton iterations, we can visually represent the solution vector y in 3D and 2D. All three examples are shown in Figures 3.8, 3.9 and 3.10.

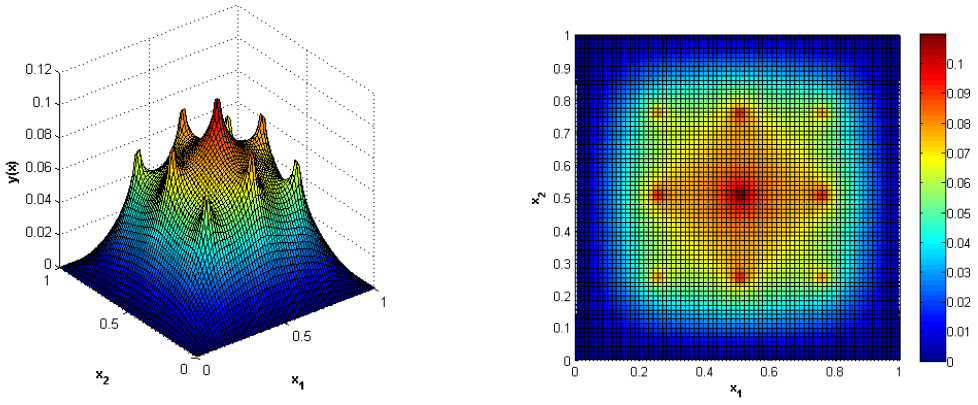


FIGURE 3.8: Example 2 - discretized solution of the semi-linear PDE 3.22 using FVM, for $m = 72$, $k = 9$, $u_i = 100$ and $E_i = 4$ for $i = 1, \ldots, k$.

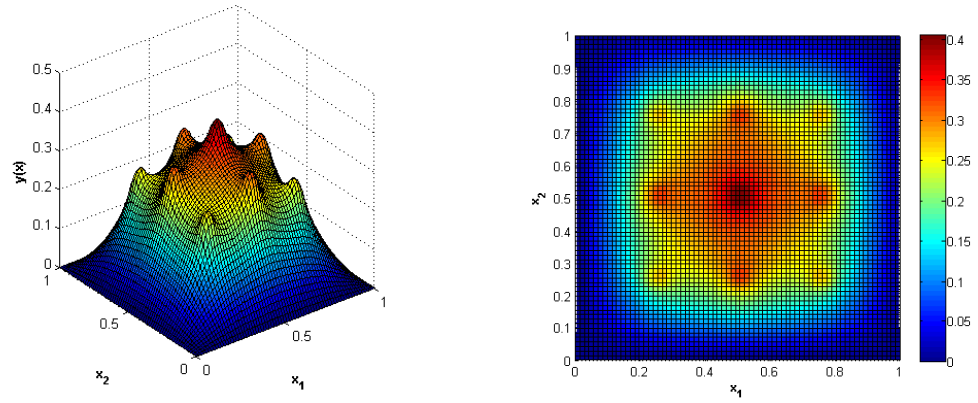


FIGURE 3.9: Example 3 - discretized solution of the semi-linear PDE 3.22 using FVM, for $m = 72$, $k = 9$, $u_i = 100$ and $E_i = 16$ for $i = 1, \ldots, k$.

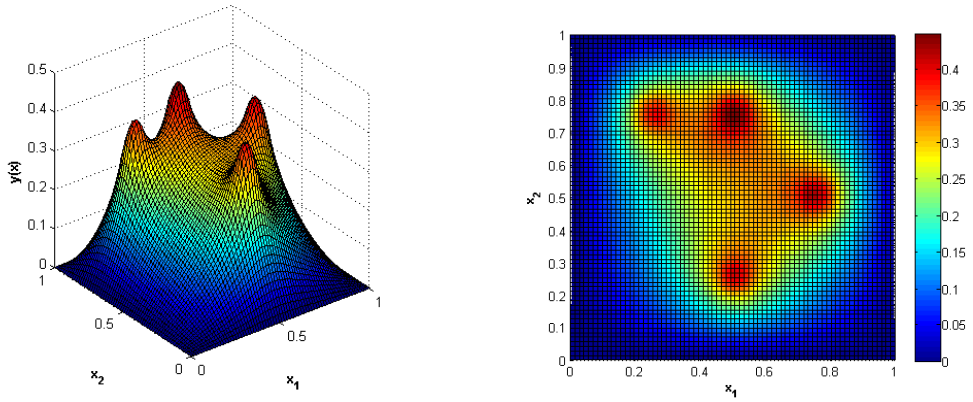


FIGURE 3.10: Example 4 - discretized solution of the semi-linear PDE 3.22 using FVM, for $m = 72$, $k = 4$, $u_i = 100$ and $E_i = 36$ for $i = 1, \ldots, k$.

The results are almost identical to the ones in linear case, demonstrated in examples 2, 3 and 4 in Chapter 3.2 (Figures 3.5, 3.6 and 3.7, respectively). This becomes obvious after investigating discretization equation 3.33 in more detail, where the differences from the linear case are marked in red. The values in red are very small due to the vector y and constant h (in this case $h^2 \approx 0.000199$). The difference is more apparent after representing the norm of the solution vector y in the *Euclidean space* (See table 3.1).

FVM	**Linear PDE:** $\|\vec{y}_1\|_{R^{m\times m}}$	**Semi-linear PDE:** $\|\vec{y}_2\|_{R^{m\times m}}$	$\|\vec{y}_1 - \vec{y}_2\|_{R^{m\times m}}$
Ex.2: $k = 9$, $E_i = 4$	3.2990	3.2983	0.0007
Ex.3: $k = 9$, $E_i = 16$	13.1616	13.1144	0.0472
Ex.4: $k = 4$, $E_i = 36$	13.6163	13.5634	0.0529

TABLE 3.1: Comparison of the solution vector y using FVM between linear and semi-linear PDEs, when $m = 72$ and $u_i = 100$ for $i = 1, \ldots, k$.

Chapter 4

Finite Element Method

In Chapter 3 Finite Volume Method was introduced. The linear and semi-linear elliptic PDEs 2.1b and 2.10b were discretized and visually represented using *MATLAB*. Newton's method was also introduced in order to linearize the semi-linear problem 2.10b. Three examples were introduced for both problems and the results were compared in Table 3.1. In Chapter 4 we will introduce and also apply another popular discretization method, which became very relevant during the last three decades: *Finite Element Method*. More information and examples of this method can be found in [4–6, 13–16].

We will start the section 4.1 with an introduction to the FEM. After that, the linear elliptic PDE 2.1b will be discretized in section 4.2, followed by the semi-linear PDE 2.10b in section 4.3. Again, we will use Newton's method to linearise a semi-linear PDE.

4.1 Introduction

Finite Element Method finds an appropriate solution of the PDE numerically, in an indirect way. The domain Ω is decomposed into smaller triangular elements called finite elements. This method is very useful for unstructured grids. In FEM we construct basis functions ψ_i, for $i = 1, \ldots, n$, where n is the number of nodes, which has finite support in such a way, that the Mass and Stiffness matrices are sparse. Therefore, it is very convenient for computer implementation.

Again, we will consider a rectangular domain $\Omega = [0, 1]^2$, governed by a PDE with homogeneous Dirichlet boundary condition. A triangular mesh is applied to the domain Ω creating finite elements with the unknown values of y placed on the nodes. This can be seen in Figure 4.1, where m is the number of nodes in either direction of the mesh

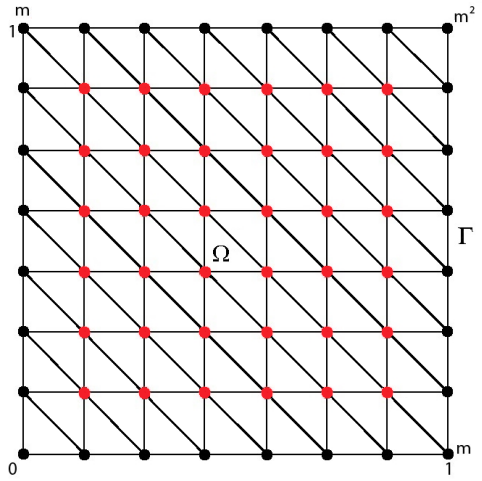


FIGURE 4.1: FEM mesh of the domain Ω

and y is represented by black and red nodes: black nodes represent boundary and the red nodes represent internal points.

4.2 Discretization of the linear boundary value problem

The same as in Chapter 3.2, we consider the linear elliptic PDE 2.1b:

$$\begin{cases} -\Delta y(x) = u(x) & x \in \Omega \\ \quad y(x) = 0 & x \in \Gamma, \end{cases} \tag{4.1}$$

where

$$u(x) = \sum_{i=1}^{k} u_i e_i(x). \tag{4.2}$$

Also, we consider $e_i(x)$ as a basis function:

$$e_i(x) = \begin{cases} 1 & \text{,in } \; x \in E_i \text{ for } i = 1, \ldots, k \\ 0 & \text{,else} \end{cases} \tag{4.3}$$

First of all, we will convert the original boundary value problem 4.1 into its weak form by multiplying both sides by a sufficiently smooth test function $\psi \in H_0^1(\Omega)$ and integrate over Ω. We are considering a standard *Galerkin approach* meaning that the test function is in the same space as the solution.

$$-\int_\Omega \Delta y \psi \, \mathrm{d}\Omega = \int_\Omega u\psi \, \mathrm{d}\Omega \tag{4.4}$$

After integration by parts, *Gauss' divergence* theorem and consideration of the boundary conditions we get the following *weak formulation*:

$$\int_\Omega \nabla y \cdot \nabla \psi \, \mathrm{d}\Omega = \int_\Omega u\psi \, \mathrm{d}\Omega. \tag{4.5}$$

Next, we perform a discretization of the weak formulation 4.5. We will write function $y(x)$ as a linear combination of the continuous piecewise linear basis functions ψ_i:

$$\begin{cases} y(x) = \sum_{i=1}^{m^2} y_i \psi_i(x) \\ \psi \to \psi_j, \quad j \in \{1, \ldots, m^2\}, \end{cases} \tag{4.6}$$

where y_i is the unknown and m^2 is the number of node points. We also choose $\psi = \psi_j$ as the basis function. By inserting 4.6 into 4.5, we get:

$$\int_\Omega \Big(\sum_{i=1}^{m^2} y_i \nabla \psi_i \Big) \nabla \psi_j \, \mathrm{d}\Omega = \int_\Omega u\psi_j \, \mathrm{d}\Omega \tag{4.7}$$

Since y_i is a scalar unknown, we can place it outside the integral:

$$\sum_{i=1}^{m^2} y_i \int_\Omega \nabla \psi_i \cdot \nabla \psi_j \, \mathrm{d}\Omega = \int_\Omega u\psi_j \, \mathrm{d}\Omega \tag{4.8}$$

This can be defined in the following form:

$$\sum_{i=1}^{m^2} S_{ij} y_i = u_j, \tag{4.9}$$

where

$$S_{ij} = \int_\Omega \nabla \psi_i \cdot \nabla \psi_j \, \mathrm{d}\Omega, \tag{4.10}$$

$$u_j = \int_\Omega u\psi_j \, \mathrm{d}\Omega. \tag{4.11}$$

Here $S_{ij}(y)$ is a *Stiffness matrix*. We want it to be sparse. Next, we insert control function 4.2 into 4.11:

$$u_j = \sum_{i=1}^{k} u_i \int_\Omega e_i \psi_j \, \mathrm{d}\Omega. \tag{4.12}$$

We want to express the right hand side using the *Mass matrix*. Hence, as $e_i(x)$ is the distribution function and ψ is the basis function, we can express $e_i(x)$ as follows:

$$e_i(x) = \sum_{t=1}^{m^2} \psi_t(x)\delta_{tE_i}, \tag{4.13}$$

where δ_{tE_i} is:

$$\delta_{tE_i} = \begin{cases} 1 & \text{if } \ t \in E_i \\ 0 & \text{else} \end{cases} \tag{4.14}$$

Then we insert $e_i(x)$ into equation 4.12:

$$\sum_{i=1}^{k} u_i \int_\Omega \Big(\sum_{t=1}^{m^2} \psi_t \delta_{tE_i} \Big) \psi_j \, \mathrm{d}\Omega = \sum_{t=1}^{m^2} \sum_{i=1}^{k} u_i \delta_{tE_i} \int_\Omega \psi_t \psi_j \, \mathrm{d}\Omega,$$

where

$$M_{tj} = \int_\Omega \psi_t \psi_j \, \mathrm{d}\Omega, \tag{4.15}$$

$$b_t = \sum_{i=1}^{k} u_i \delta_{tE_i} \tag{4.16}$$

Here M_{tj} is the Mass matrix and we want it to be sparse. Finally, we end up with the following form:

$$\sum_{i=1}^{m^2} S_{ij} y_i = \sum_{t=1}^{m^2} M_{tj} b_t \tag{4.17}$$

Next step is to find the global element matrices S_{ij} and M_{tj} using the local element matrices:

$$S_{ij} = \sum_{p=1}^{n_e} \int_{e_p} \nabla\psi_i \cdot \nabla\psi_j \, \mathrm{d}\Omega = \sum_{p=1}^{n_e} S_{ij}^{e_p}, \tag{4.18}$$

$$M_{tj} = \sum_{p=1}^{n_e} \int_{e_p} \psi_t \psi_j \, \mathrm{d}\Omega = \sum_{p=1}^{n_e} M_{tj}^{e_p}. \tag{4.19}$$

Here n_e is the number of elements and p represents the current element (Figure 4.2).

To build the global Stiffness and Mass matrices, we first define the transformation from the element in given domain to the standard element and then we obtain local Stiffness and Mass matrices for each element. Afterwards, we obtain global Stiffness and Mass matrices from local matrices using the mapping. This is followed by solving the

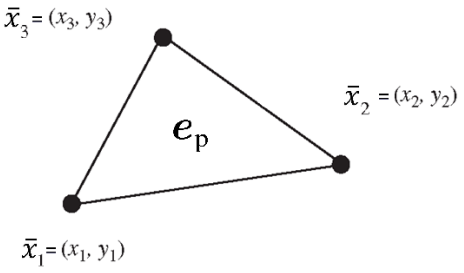


FIGURE 4.2: Finite element triangle e_p with three y nodes represented by $\bar{x}_1, \bar{x}_2$ and $\bar{x}_3$

sparse linear system described in equation 4.17 to obtain the unknown y_i. Then we can construct the original solution $y(x)$ using equation 4.6. After implementing this in MATLAB, we get the following results for different examples:

Again, as in Chapter 3.2, we consider a simple Example 1, where $k = 1$, $u_i = 1$ and $E_i = \Omega$ for $i = 1, \ldots, k$. Hence, the control function $u(x) = 1$ for the whole domain Ω is:

$$\begin{cases} -\Delta y(x) = u(x) & x \in \Omega \\ y(x) = 0 & x \in \Gamma \end{cases} \tag{4.20}$$

By discretizing this PDE using linear system 4.17 we obtain the following results for y in Figure 4.3, where $m = 30$.

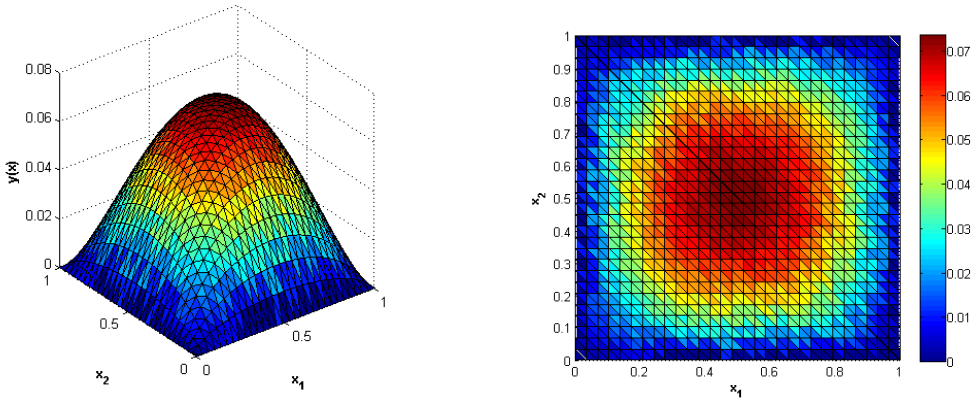


FIGURE 4.3: Example 1 - discretized solution of the linear PDE 4.20 using FEM, for $m = 30$, $k = 1$, $u_i = 1$ and $E_i = \Omega$ for $i = 1, \ldots, k$.

Next, let us consider the three previous examples, Ex.2, Ex.3 and Ex.4, described in Chapters 3.2 and 3.4, for linear PDE 4.1 using its linear system 4.17. We discretize this PDE using FEM on MATLAB and plot the solution vector y. Examples 2, 3 and 4 are represented in Figures 4.4, 4.5 and 4.6.

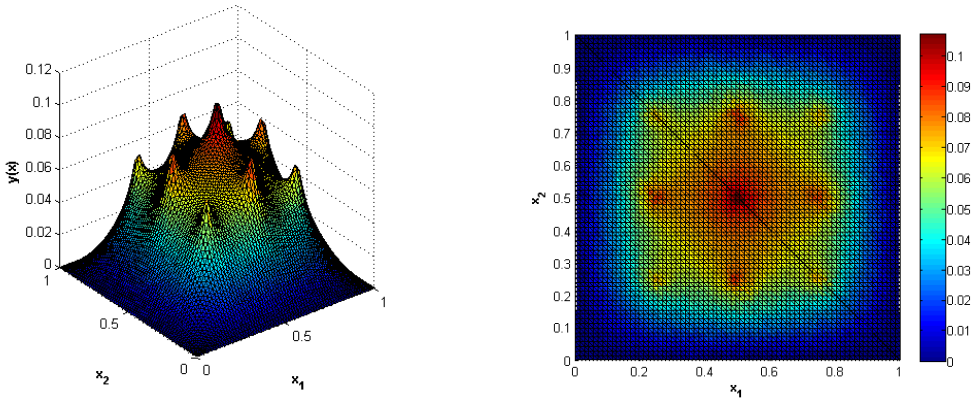


FIGURE 4.4: Example 2 - discretized solution of the linear PDE 4.1 using FEM, for $m = 72$, $k = 9$, $u_i = 100$ and $E_i = 4$ for $i = 1, \ldots, k$.

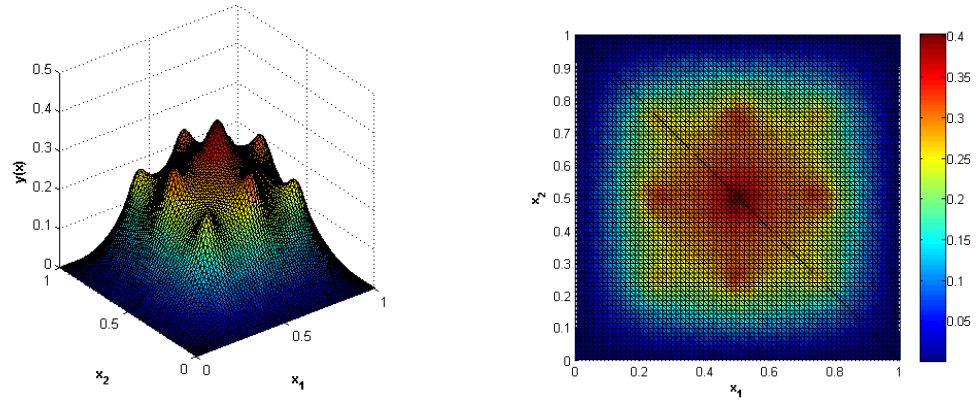


FIGURE 4.5: Example 3 - discretized solution of the linear PDE 4.1 using FEM, for $m = 72$, $k = 9$, $u_i = 100$ and $E_i = 16$ for $i = 1, \ldots, k$.

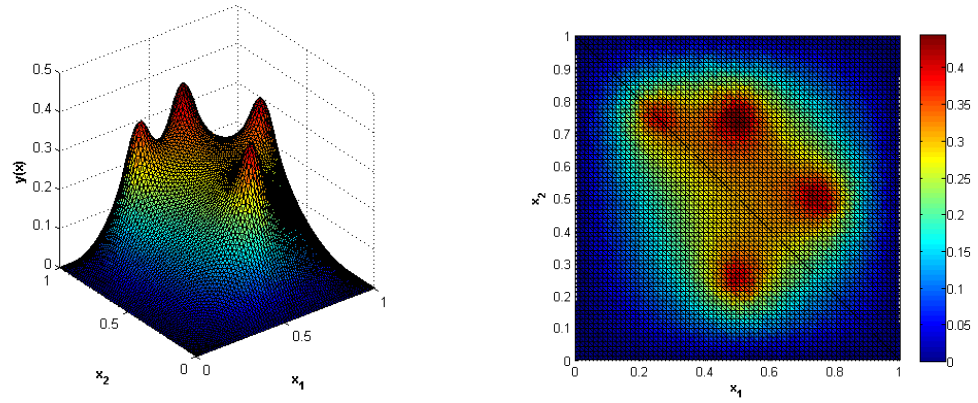


FIGURE 4.6: Example 4 - discretized solution of the linear PDE 4.1 using FEM, for $m = 72$, $k = 4$, $u_i = 100$ and $E_i = 36$ for $i = 1, \ldots, k$.

4.3 Discretization of the semi-linear boundary value problem

Let us consider a semi-linear elliptic PDE 2.10b in the same way as in Chapter 3.4:

$$\begin{cases} -\Delta y(x) + d(x,y) = u(x) & x \in \Omega \\ y(x) = 0 & x \in \Gamma, \end{cases} \tag{4.21}$$

where

$$u(x) = \sum_{i=1}^{k} u_i e_i(x). \tag{4.22}$$

Again, we use $e_i(x)$ as a basis function:

$$e_i(x) = \begin{cases} 1 & ,x \in E_i \text{ for } i = 1, \ldots, k \\ 0 & ,\text{else} \end{cases} \tag{4.23}$$

As the PDE 4.21 is semi-linear, we will be using *Newton's method* to linearise it. We will start by finding the initial vector y^0 by solving the PDE 4.21 without the semi-linear part $d(x,y)$:

$$\begin{cases} -\Delta y^0(x) = u(x) & x \in \Omega \\ y^0(x) = 0 & x \in \Gamma. \end{cases} \tag{4.24}$$

PDE 4.24 becomes the same as linear PDE 4.1 in Chapter 4.2, for which we have already obtained vector $y^0 = y$. Next step would be to apply the *Newton's method* to the PDE 4.21 in the same way as described in Chapter 3.3. After doing so, we get the following equation (3.21):

$$-\Delta y^{n+1} + d'(y^n)y^{n+1} = u - d(y^n) + d'(y^n)y^n, \tag{4.25}$$

where y^n is the previous known value and y^{n+1} is unknown. Again, lets take $d(y) = y^3$, which leads to the following linear PDE:

$$\begin{cases} -\Delta y^{n+1} + 3(y^n)^2 y^{n+1} = u + 2(y^n)^3 & x \in \Omega \\ y^{n+1} = 0 & x \in \Gamma. \end{cases} \tag{4.26}$$

Now, we will apply the FEM discretization. First of all, we will rephrase the newly obtained boundary value problem 4.26 into its weak form by multiplying both sides by a sufficiently smooth test function $\psi \in H_0^1(\Omega)$ and integrating over Ω. Again, we consider a standard *Galerkin approach*, therefore, the test function is in the same space as the solution.

$$-\int_\Omega \Delta y^{n+1}\psi \,\mathrm{d}\Omega + \int_\Omega 3(y^n)^2 y^{n+1}\psi \,\mathrm{d}\Omega = \int_\Omega u\psi \,\mathrm{d}\Omega + \int_\Omega 2(y^n)^3\psi \,\mathrm{d}\Omega \tag{4.27}$$

After integration by parts, *Gauss' divergence theorem* and consideration of the boundary conditions we get the weak formulation:

$$\int_\Omega \nabla y^{n+1}\cdot\nabla\psi \,\mathrm{d}\Omega + 3(y^n)^2\int_\Omega y^{n+1}\psi \,\mathrm{d}\Omega = \int_\Omega u\psi \,\mathrm{d}\Omega + \int_\Omega 2(y^n)^3\psi \,\mathrm{d}\Omega \tag{4.28}$$

Next, we will express function $y(x)$ as a linear combination of the continuous piecewise linear basis functions ψ_i:

$$\begin{cases} y(x)^{n+1} = \sum_{i=1}^{m^2} y_i\psi_i(x) \\ \psi \to \psi_j, \quad j \in \{1,\dots,n\}, \end{cases} \tag{4.29}$$

where y_i is the unknown and m^2 is the number of the node points. By inserting 4.29 into 4.28, we get:

$$\int_\Omega \Big(\sum_{i=1}^{m^2} y_i\nabla\psi_i\Big)\nabla\psi_j \,\mathrm{d}\Omega + 3(y^n)^2\int_\Omega \Big(\sum_{i=1}^{m^2} y_i\psi_i\Big)\psi_j \,\mathrm{d}\Omega = \int_\Omega u\psi_j \,\mathrm{d}\Omega + \int_\Omega 2(y^n)^3\psi_j \,\mathrm{d}\Omega$$

Since, y_i is the scalar unknown, we can place it outside the integral:

$$\sum_{i=1}^{m^2} y_i\int_\Omega \nabla\psi_i\cdot\nabla\psi_j \,\mathrm{d}\Omega + 3(y^n)^2\sum_{i=1}^{m^2} y_i\int_\Omega \psi_i\psi_j \,\mathrm{d}\Omega = \int_\Omega u\psi_j \,\mathrm{d}\Omega + \int_\Omega 2(y^n)^3\psi_j \,\mathrm{d}\Omega$$

Which can be written in a shorter form:

$$\sum_{i=1}^{m^2} S_{ij}y_i + 3(y^n)^2\sum_{i=1}^{m^2} M_{ij}y_i = u_j + \int_\Omega 2(y^n)^3\psi_j \,\mathrm{d}\Omega, \tag{4.30}$$

where

$$S_{ij} = \int_\Omega \nabla\psi_i\cdot\nabla\psi_j \,\mathrm{d}\Omega, \tag{4.31}$$

$$M_{ij} = \int_\Omega \psi_i\psi_j \,\mathrm{d}\Omega, \tag{4.32}$$

$$u_j = \int_\Omega u\psi_j \,\mathrm{d}\Omega. \tag{4.33}$$

Here $S_{ij}(y)$ and $M_{ij}(y)$ are called Stiffness and Mass matrices, respectively. We want to express the right hand side using *Mass matrices*. Hence, we insert the control function $u(x)$ into 4.33:

$$u_j = \sum_{i=1}^{k} u_i \int_\Omega e_i \psi_j \, \mathrm{d}\Omega. \tag{4.34}$$

As $e_i(x)$ is the distribution function and ψ is the basis function, we can express $e_i(x)$ as follows:

$$e_i(x) = \sum_{t=1}^{m^2} \psi_t(x) \delta_{tE_i}, \tag{4.35}$$

where δ_{tE_i} is:

$$\delta_{tE_i} = \begin{cases} 1 & \text{if } \ t \in E_i \\ 0 & \text{else .} \end{cases} \tag{4.36}$$

Then we insert $e_i(x)$ into equation 4.34:

$$\sum_{i=1}^{k} u_i \int_\Omega \Big(\sum_{t=1}^{m^2} \psi_t \delta_{tE_i} \Big) \psi_j \, \mathrm{d}\Omega = \sum_{t=1}^{m^2} \sum_{i=1}^{k} u_i \delta_{tE_i} \int_\Omega \psi_t \psi_j \, \mathrm{d}\Omega,$$

where

$$M_{tj} = \int_\Omega \psi_t \psi_j \, \mathrm{d}\Omega, \tag{4.37}$$

$$b_t = \sum_{i=1}^{k} u_i \delta_{tE_i} \tag{4.38}$$

Here M_{tj} is also the Mass matrix. Therefore, we end up with the following form:

$$\sum_{i=1}^{m^2} S_{ij} y_i + 3(y^n)^2 \sum_{i=1}^{m^2} M_{ij} y_i = \sum_{t=1}^{m^2} M_{tj} b_t + \int_\Omega 2(y^n)^3 \psi_j \, \mathrm{d}\Omega. \tag{4.39}$$

However, we still have $2(y^n)^3$ on the right-hand side. Lets call it $\phi(x)$:

$$\phi(x) = 2(y^n)^3 \tag{4.40}$$

We can repeat the procedure and express $\phi(x)$ as a linear combination of some basis functions:

$$\phi(x) = \sum_{i=1}^{m^2} \phi_i \psi_i(x). \tag{4.41}$$

We insert $\phi(x)$ into 4.39:

$$\sum_{i=1}^{m^2} S_{ij}y_i + 3(y^n)^2 \sum_{i=1}^{m^2} M_{ij}y_i = \sum_{t=1}^{m^2} M_{tj}b_t + \sum_{i=1}^{m^2} \phi_i \int_\Omega \psi_i\psi_j \, \mathrm{d}\Omega. \tag{4.42}$$

Now, we express the second term using the *Mass matrix*. Hence, the final sparse linear system is of the following form:

$$\sum_{i=1}^{m^2} S_{ij}y_i + 3(y^n)^2 \sum_{i=1}^{m^2} M_{ij}y_i = \sum_{t=1}^{m^2} M_{tj}b_t + 2(y^n)^3 \sum_{i=1}^{m^2} M_{ij} \tag{4.43}$$

Next step is to find the global element matrices S_{ij} and M_{ij} using local element matrices:

$$S_{ij} = \sum_{p=1}^{n_e} \int_{e_p} \nabla\psi_i \cdot \nabla\psi_j \, \mathrm{d}\Omega = \sum_{p=1}^{n_e} S_{ij}^{e_p}, \tag{4.44}$$

$$M_{ij} = \sum_{p=1}^{n_e} \int_{e_p} \psi_i\psi_j \, \mathrm{d}\Omega = \sum_{p=1}^{n_e} M_{ij}^{e_p}. \tag{4.45}$$

Here n_e is the number of elements and p represents the current element (Figure 4.2).

We prepare and solve the sparse linear system 4.43 as described in Chapter 4.2. Then we can construct the original solution $y(x)$ using equation 4.29. After implementing this in MATLAB, we get the following results.

Let us consider the three previous examples, Ex.2, Ex.3 and Ex.4, described in Chapters 3.2 and 3.4, for the semi-linear PDE 4.21 using its linear system 4.43. We discretize this PDE using FEM on MATLAB and after three Newton iterations we get the solution vector y. Examples 2, 3 and 4 are represented in Figures 4.7, 4.8 and 4.9, respectively.

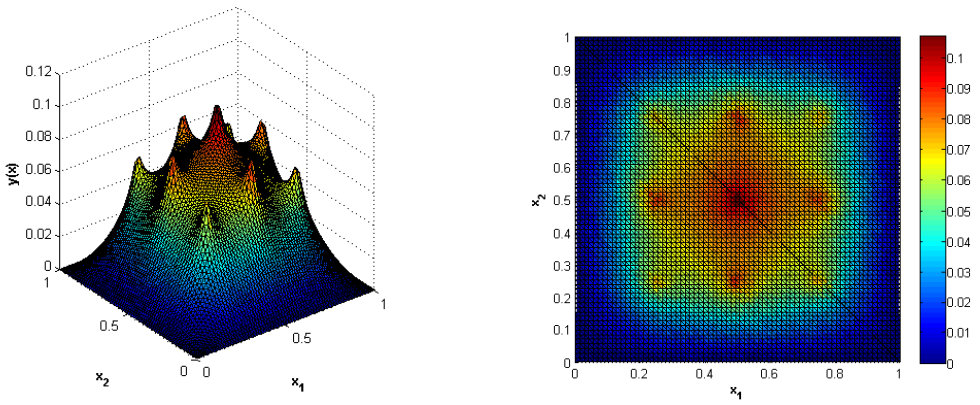


FIGURE 4.7: Example 2 - discretized solution of the semi-linear PDE 4.21 using FEM, for $m = 72$, $k = 9$, $u_i = 100$ and $E_i = 4$ for $i = 1, \ldots, k$.

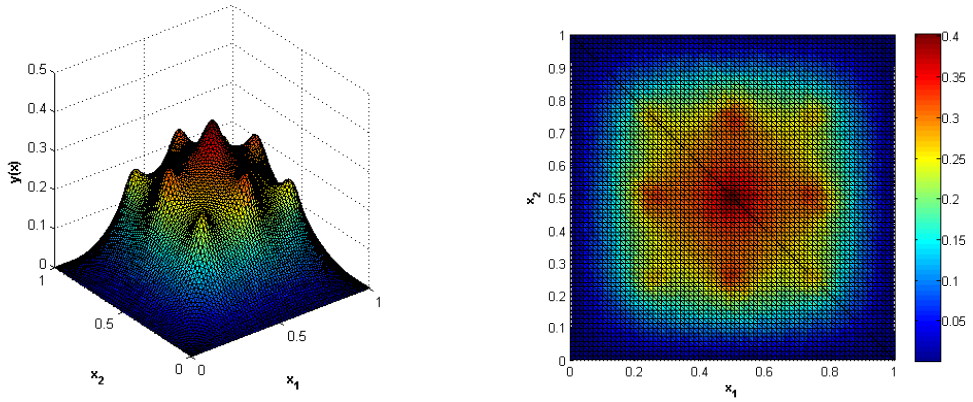


FIGURE 4.8: Example 3 - discretized solution of the semi-linear PDE 4.21 using FEM, for $m = 72$, $k = 9$, $u_i = 100$ and $E_i = 16$ for $i = 1, \ldots, k$.

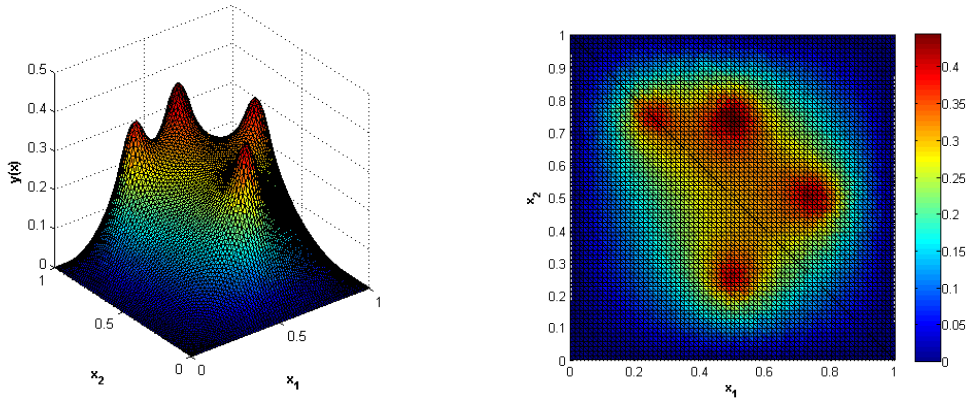


FIGURE 4.9: Example 4 - discretized solution of the semi-linear PDE 4.21 using FEM, for $m = 72$, $k = 4$, $u_i = 100$ and $E_i = 36$ for $i = 1, \ldots, k$.

Again, the results are almost identical to those of Examples 2, 3 and 4 in the linear case (Figures 4.4, 4.5 and 4.6, respectively) in Chapter 4.2. The differences are more apparent after representing the norm of the solution vector y in the *Euclidean space* (See table 4.1).

FEM	**Linear PDE:** $\|\vec{y}_1\|_{R^{m\times m}}$	**Semi-linear PDE:** $\|\vec{y}_2\|_{R^{m\times m}}$	$\|\vec{y}_1 - \vec{y}_2\|_{R^{m\times m}}$
Ex.2: $k = 9$, $E_i = 4$	3.2960	3.2953	0.0007
Ex.3: $k = 9$, $E_i = 16$	13.1510	13.1040	0.047
Ex.4: $k = 4$, $E_i = 36$	13.6038	13.5511	0.0527

TABLE 4.1: Comparison of the solution vector y using FEM between linear and semi-linear PDEs, when $m = 72$ and $u_i = 100$ for $i = 1, \ldots, k$.

Chapter 5

Optimization methods for optimal control problems

In Chapter 4 *Finite Element Method* was introduced. The linear and semi-linear elliptic PDEs 2.1b and 2.10b, respectively were discretized and visually represented using *MATLAB*.The same three examples were introduced for each of the two problems and the results were compared in Table 4.1. As in Chapter 3, the semi-linear PDE was first linearised using *Newton's method*. In Chapter 5 we will introduce and apply different optimization methods for both optimal control problems 2.1 and 2.10. We will test these optimization methods for optimal control problems using *Finite Volume* and *Finite Element Methods* and compare the results. There is a lot of information on numerical optimization. As an example, the reader can be refereed to [5, 11, 12].

In section 5.1 we will start with preparation for optimization methods. After that, in section 5.2 we will transform optimal control problem into *reduced quadratic optimization problem*. Then we will start introducing and comparing optimization techniques for linear and semi-linear optimal control problems 2.1 and 2.10: *quadprog* in section 5.2.1, *fmincon* in section 5.2.2, whereas in sections 5.2.3 and 5.2.4 we will introduce *fmincon* with supplied gradient and *projected gradient* methods. In section **??** we will draw a conclusion from these methods.

5.1 Preparation

Let us consider two optimal control problems 2.1 and 2.10 presented in Chapter 2 as $P1$ and $P2$, respectively.

$$P1 = \begin{cases} min\ J(y,u) := \frac{1}{2}\|y - y_\Omega\|^2_{L^2(\Omega)} + \frac{\lambda}{2}\|u\|^2_{L^2(\Omega)} & \\ -\Delta y(x) = u(x) & \text{in } \Omega \\ \quad y(x) = 0 & \text{on } \Gamma \\ u_a(x) \leq u(x) \leq u_b(x) & u \in U_{ad} \end{cases} \tag{5.1}$$

$$P2 = \begin{cases} min\ J(y,u) := \frac{1}{2}\|y - y_\Omega\|^2_{L^2(\Omega)} + \frac{\lambda}{2}\|u\|^2_{L^2(\Omega)} & \\ -\Delta y(x) + d(x,y) = u(x) & \text{in } \Omega \\ \quad y(x) = 0 & \text{on } \Gamma \\ u_a(x) \leq u(x) \leq u_b(x) & u \in U_{ad} \end{cases} \tag{5.2}$$

Here the control is situated on the Ω. For both cases we investigate a special ansatz 5.4 of the control function, where the control function $u(x)$ is a linear combination of some basis functions

$$u(x) = \sum_{i=1}^{k} u_i e_i(x), \tag{5.3}$$

with real values control parameter u_i and finitely many given distribution functions $e_i : \Omega \to \mathbb{R}$ (Figure 2.1). Moreover, we have the *control constraints* to the control u_i

$$u_{\min} \leq u_i \leq u_{\max} \quad u_i \in U_{ad,k}, \tag{5.4}$$

where $u_{\min} < u_{\max}$ are given real numbers. We take $e_i(x)$ as a basis function

$$e_i(x) = \begin{cases} 1 & , x \in E_i \text{ for } i = 1, \ldots, k \\ 0 & , \text{else} \end{cases} \tag{5.5}$$

We have previously discretized PDEs of $P1$ and $P2$ using FVM and FEM to obtain the solution y. Next step would be to test different optimization methods for these problems to find optimal $\bar{u}$ and $\bar{y}$. For that reason, we will use the following general settings for all four different techniques: $m = 96$, $k = 9$, initial control $u_i = 1$ and $E_i = 8 \times 8$ for $i = 1, \ldots, k$. For semi-linear case we will consider $d(x,y) = y^3$ and we will always start with linearisation of the equation using *Newton's method*.

5.2 Different optimization techniques for optimal control problems

We will use the solution operator Λ of PDE introduced in Chapter 2 to derive the *reduced optimization problem* for the linear case $P1$.

$$y(x) = \Lambda u(x) \tag{5.6}$$

$$\Lambda u = Su = S\Big(\sum_{i=1}^{k} u_i e_i\Big) \tag{5.7}$$

$$J(y,u) = J(Su,u) = f(\vec{u}) \tag{5.8}$$

By inserting 5.8 into cost functional, we get the reduced objective functional

$$\begin{aligned}\min_{\vec{u}\in\vec{U}_{ad,k}} f(\vec{u}) &= \frac{1}{2}\|Su - y_\Omega\|^2_{L^2(\Omega)} + \frac{\lambda}{2}\|u\|^2_{L^2(\Omega)} \\ &= \frac{1}{2}\|\Lambda\vec{u} - y_\Omega\|^2_{L^2(\Omega)} + \frac{\lambda}{2}\|\sum_{i=1}^{k} u_i e_i\|^2_{L^2(\Omega)}\end{aligned} \tag{5.9}$$

We want to transform problem $P1$ into reduced quadratic optimization problem. We insert 5.3 into 5.6 to get the following functions:

$$y_i(x) = \Lambda e_i(x) \qquad i = 1,\dots,k, \tag{5.10}$$

where functions $y_i(x)$ are the solution to the problems

$$\begin{cases} -\Delta y = e_i & ,in\ \ \Omega \\ \quad y = 0 & ,on\ \Gamma \end{cases}$$

After solving k PDE's, $y = \Lambda u$ can be found by superposition:

$$y = \sum_{i=1}^{k} u_i y_i(x) \tag{5.11}$$

Now the quadratic optimization problem can be established by inserting 5.3 and 5.11 into reduced cost functional

$$\begin{aligned}\min_{\vec{u}\in\vec{U}_{ad,k}} f(\vec{u}) &= \frac{1}{2}\Big\|\sum_{i=1}^{k} u_i y_i - y_\Omega\Big\|^2_{L^2(\Omega)} + \frac{\lambda}{2}\Big\|\sum_{i=1}^{k} u_i e_i\Big\|^2_{L^2(\Omega)} \\ &= \frac{1}{2}\|y_\Omega\|^2_{L^2(\Omega)} - \sum_{i=1}^{k} u_i (y_\Omega, y_i)_{L^2(\Omega)} + \frac{1}{2}\sum_{i,j=1}^{k} u_i u_j (y_i, y_j)_{L^2(\Omega)} \\ &+ \frac{\lambda}{2}\sum_{i,j=1}^{k} u_i u_j (e_i, e_j)_{L^2(\Omega)} \end{aligned} \tag{5.12}$$

The first term of the function is constant and we can remove it, as it does not influence the optimization:

$$\begin{aligned}\min_{\vec{u}\in\vec{U}_{ad,k}} f(\vec{u}) = \frac{1}{2}\sum_{i,j=1}^{k} u_i u_j (y_i, y_j)_{L^2(\Omega)} - \sum_{i=1}^{k} u_i (y_\Omega, y_i)_{L^2(\Omega)} \\ + \frac{\lambda}{2}\sum_{i,j=1}^{k} u_i u_j (e_i, e_j)_{L^2(\Omega)}\end{aligned} \tag{5.13}$$

We can reduce the equation 5.13 by choosing the following notations:

$$a_i = (y_\Omega, y_i)_{L^2(\Omega)} = \int_\Omega y_\Omega(x) y_i(x)\,\mathrm{d}\Omega \tag{5.14}$$

$$C_{ij} = (y_i, y_j)_{L^2(\Omega)} = \int_\Omega y_i(x) y_j(x)\,\mathrm{d}\Omega \tag{5.15}$$

$$D_{ij} = (e_i, e_j)_{L^2(\Omega)} = \int_\Omega e_i(x) e_j(x)\,\mathrm{d}\Omega \tag{5.16}$$

$$H = C + \lambda D \tag{5.17}$$

Here $C, D, H \in \mathbb{R}^{k\times k}$ and $\vec{a} \in \mathbb{R}^{k\times 1}$. Then using these notations, we can transform problem 5.13 into the shorter form:

$$\min_{\vec{u}\in\vec{U}_{ad,k}} f(\vec{u}) = \min_{\vec{u}\in\vec{U}_{ad,k}} \left\{\frac{1}{2}\vec{u}^T H \vec{u} - \vec{a}^T \vec{u}\right\} \tag{5.18}$$

This reduced quadratic optimization problem of $P1$ can be solved using different optimization methods, which we will discuss in the following sections. For the semi-linear optimal control problem $P2$ we will use reduced cost functional 5.9.

5.2.1 Optimization using *quadprog*

Let us consider reduced quadratic optimization problem 5.18. This problem can be easily solved using code *quadprog* in MATLAB.

First, we discretize the optimal control problem $P1$ and then use *quadprog* to solve the quadratic optimization problem. The aim here is to find the optimal control $\vec{u}$, so that $\bar{y}$ is the closest to the desired state y_Ω.

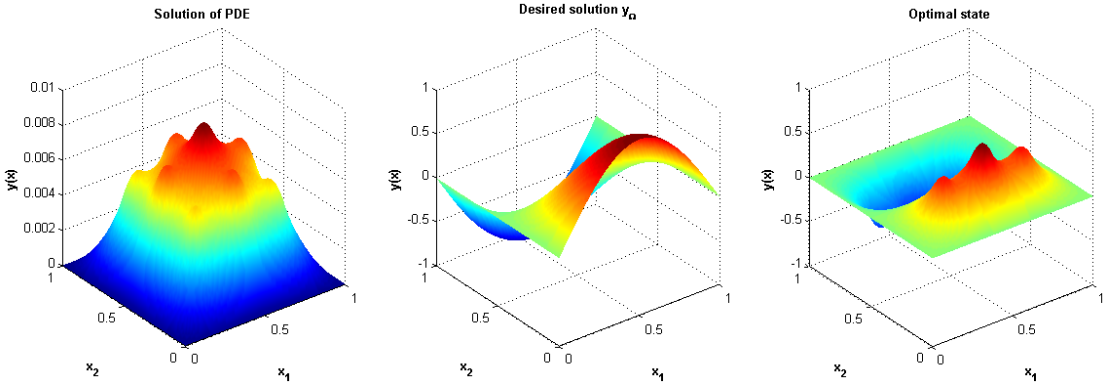


FIGURE 5.1: Optimization results of the optimal control problem $P1$ using FEM, together with *quadprog* code in 3D: $m = 96$, $k = 9$, initial control $u_i = 1$ and $E_i = 8 \times 8$ for $i = 1, \ldots, k$

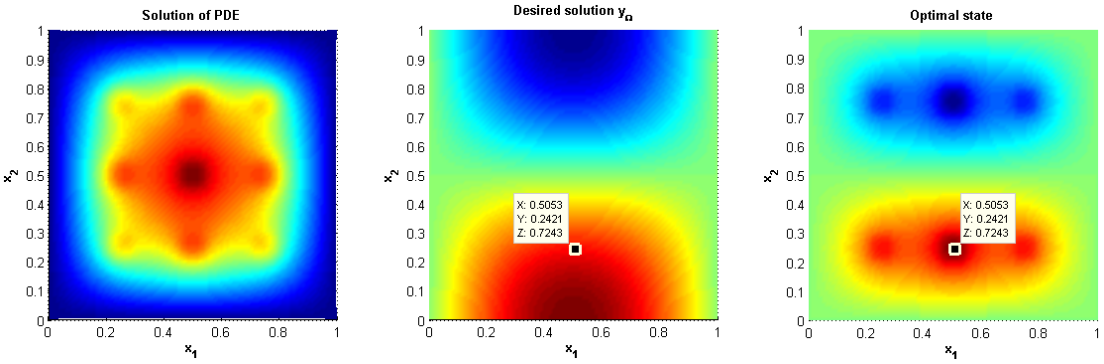


FIGURE 5.2: Optimization results of the optimal control problem $P1$ using FEM, together with *quadprog* code in 2D: $m = 96$, $k = 9$, initial control $u_i = 1$ and $E_i = 8 \times 8$ for $i = 1, \ldots, k$

In Figures 5.1 and 5.2 we can see three plots representing different stages of the solution of $P1$. First plot represents the solution y of the PDE. Second plot shows the chosen *desired solution* y_Ω, which is the solution that we would like to achieve at the control points. Third plot represents the optimal solution state $\bar{y}$, which we get by finding the optimal control vector $\vec{u}$. The difference between using this method together with FVM or FEM is seen in the Table 5.1.

quadprog	FVM	FEM
CPU - time	1.57284	2.958231
$f(\vec{u})$	-0.0509	-0.0508

TABLE 5.1: Results of the *quadprog* optimization of the problem $P1$ using FVM and FEM discretization

As it is seen in Figures 5.2, the optimization technique provides the accurate result, because the optimal state at the chosen point is exactly the same as the desired solution y_Ω. However, *quadprog* does not work with semi-linear problems, hence we can not use it for problem $P2$.

5.2.2 Optimization using *fmincon*

Quadratic optimization problem 5.18 can be easily optimized using another MATLAB code *fmincon*, that finds the constrained minimum starting at an initial guess $\vec{u}^0$.

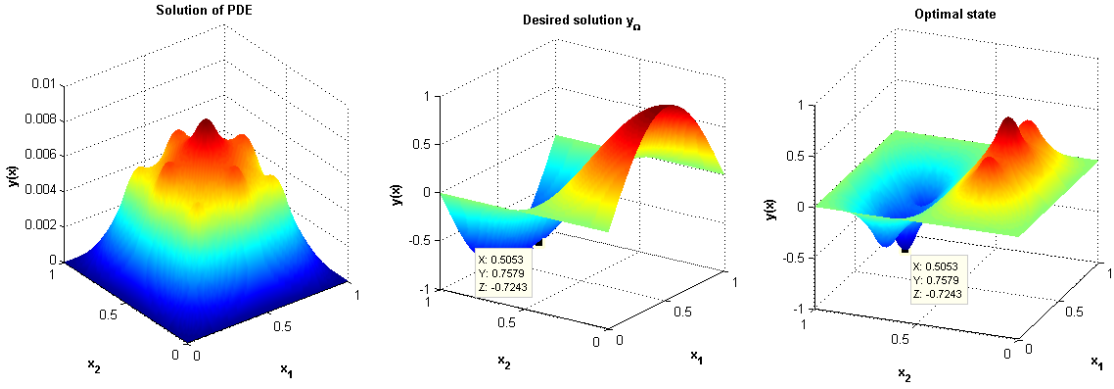


FIGURE 5.3: Optimization results of the optimal control problem $P1$ using FVM together with *fmincon* code in 3D: $m = 96$, $k = 9$, initial control $u_i = 1$ and $E_i = 8 \times 8$ for $i = 1, \ldots, k$

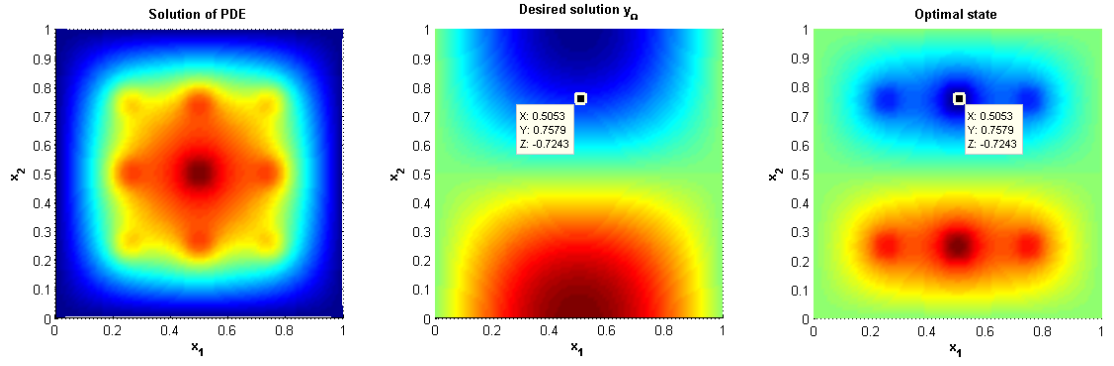


FIGURE 5.4: Optimization results of the optimal control problem $P1$ using FVM together with *fmincon* code in 2D: $m = 96$, $k = 9$, initial control $u_i = 1$ and $E_i = 8 \times 8$ for $i = 1, \ldots, k$

In Figures 5.3 and 5.4 we can see the results of $P1$ using FVM together with *fmincon* code. For semi-linear problem $P2$ we will use cost functional 5.9 to achieve optimization.

The results of different discretization techniques can be seen in the Table 5.2. We get almost identical results for linear and semi-linear problems, which means that the methods are working correctly. We can notice that FEM is slower than FVM, as the number of calculations is greater in FEM. This should result in a more accurate solution. Also, we can see a huge difference in computational time between solving linear and semi-linear optimal control problems. Therefore, it is not suitable for complicated problems, as it might take too long to run.

fmincon	**FVM**		**FEM**	
	Linear	Semi-linear	Linear	Semi-linear
CPU-time	2.17389	299.523994	3.553294	945.401196
$f(\vec{u})$	-0.0509	0.0717	-0.0508	0.0717

TABLE 5.2: Results of the *fmincon* optimization of the problems $P1$ and $P2$ using FVM and FEM discretization

In general, *fmincon* is a slow method, because it computes gradient of objective functional at every step. However, we can reduce the computational time by supplying gradient to *fmincon*. Hence, the next step is to find $\nabla f(\vec{u})$.

5.2.3 Optimization using *fmincon* with supplied gradient

We start by considering the reduced cost functional introduced in Chapter 5.2.

$$f(\vec{u}) = \frac{1}{2}\|\Lambda\vec{u} - y_\Omega\|^2_{L^2(\Omega)} + \frac{\lambda}{2}\|\sum_{i=1}^{k} u_i e_i\|^2_{L^2(\Omega)} \qquad \vec{u} \in \vec{U}_{ad,k} \tag{5.19}$$

On the right-hand side we have

$$\begin{aligned}\frac{\lambda}{2}\|\sum_{i=1}^{k} u_i e_i\|^2_{L^2(\Omega)} &= \frac{\lambda}{2}\Big(\sum_{i=1}^{k} u_i e_i, \sum_{i=1}^{k} u_j e_j\Big)_{L^2(\Omega)} \\ &= \frac{\lambda}{2}\sum_{i,j=1}^{k} u_i u_j (e_i, e_j)_{L^2(\Omega)} = \frac{\lambda}{2}(\vec{u}, D\vec{u})_{\mathbb{R}^k}\end{aligned} \tag{5.20}$$

where

$$D_{ij} = (e_i, e_j)_{L^2(\Omega)} = \int_\Omega e_i(x)e_j(x)\,\mathrm{d}\Omega. \tag{5.21}$$

Hence

$$\nabla \frac{\lambda}{2}(\vec{u}, D\vec{u})_{\mathbb{R}^k} = \lambda D\vec{u}. \tag{5.22}$$

Then, by using gradient 5.22 we can write derivative of f in the direction h, where h represents the rate of change

$$f'(\vec{u})\vec{h} = (\Lambda^*(\Lambda\vec{u} - y_\Omega), \vec{h})_{\mathbb{R}^k} + (\lambda D\vec{u}, \vec{h})_{\mathbb{R}^k}, \tag{5.23}$$

where the operator Λ^* still needs to be found. According to [5], operator Λ^* can be given by the relation

$$(\Lambda\vec{u}, z)_{L^2(\Omega)} = (\vec{u}, \Lambda^* z)_{\mathbb{R}^k} \qquad \forall z \in L^2(\Omega), \quad \forall \vec{u} \in \mathbb{R}^k. \tag{5.24}$$

Then, by inserting 5.7 into 5.24 we get

$$\begin{aligned}(\Lambda\vec{u}, z)_{L^2(\Omega)} &= (S\Big(\sum_{i=1}^{k} u_i e_i\Big), z)_{L^2(\Omega)} = \Big(\sum_{i=1}^{k} u_i e_i, S^* z\Big)_{L^2(\Omega)} \\ &= \sum_{i=1}^{k} u_i (e_i, S^* z)_{L^2(\Omega)} = \left(\vec{u}, \begin{bmatrix} (p, e_1)_{L^2(\Omega)} \\ \vdots \\ (p, e_k)_{L^2(\Omega)} \end{bmatrix}\right)_{\mathbb{R}^k},\end{aligned} \tag{5.25}$$

where $S^* : L^2(\Omega) \to L^2(\Omega)$ is the adjoint operator and $S^* z = p$ is a solution to the adjoint problem [5]

$$\begin{cases} -\Delta p = z & \text{in } \Omega \\ \quad p = 0 & \text{on } \Gamma. \end{cases} \tag{5.26}$$

Then

$$\Lambda^* z = \begin{bmatrix} (p, e_1)_{L^2(\Omega)} \\ \vdots \\ (p, e_k)_{L^2(\Omega)} \end{bmatrix} \tag{5.27}$$

Finally, using 5.22 and 5.27 we find the gradient

$$\nabla f(\vec{u}) = \begin{bmatrix} (p, e_1)_{L^2(\Omega)} \\ \vdots \\ (p, e_k)_{L^2(\Omega)} \end{bmatrix} + \lambda D\vec{u} \qquad \vec{u} \in \mathbb{R}^k \tag{5.28}$$

We get the adjoint equation for $P1$

$$\begin{cases} -\Delta p = y - y_\Omega & \text{in } \Omega \\ p = 0 & \text{on } \Gamma. \end{cases} \tag{5.29}$$

We have previously obtained the adjoint equation for $P2$ in Chapter 2.5

$$\begin{cases} -\Delta p + d_y(x, \bar{y})p = y - y_\Omega & \text{in } \Omega \\ p = 0 & \text{on } \Gamma \end{cases} \tag{5.30}$$

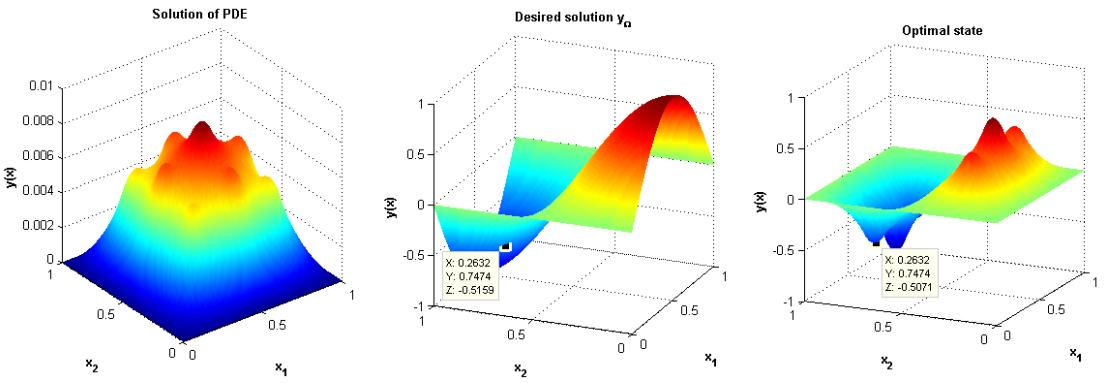


FIGURE 5.5: Optimization results of the optimal control problem $P2$ using FEM together with *fmincon* with supplied gradient code in 3D: $m = 96$, $k = 9$, initial control $u_i = 1$ and $E_i = 8 \times 8$ for $i = 1, \ldots, k$

In Figure 5.5 we can see the optimal state $\bar{y}$ of P2 by using *fmincon* together with FEM. Again, at one of the points we can see that the actual solution is following the desired solution very closely.

fmincon + ∇	**FVM**		**FEM**	
	Linear	Semi-linear	Linear	Semi-linear
CPU-time	1.896378	50.087365	3.42329	516.93376
$f(\vec{\bar{u}})$	-0.0505	0.0717	-0.0505	0.0717

TABLE 5.3: Results of the *fmincon* optimization with supplied gradient of the problems $P1$ and $P2$ using FVM and FEM discretization

It can be seen from Table 5.3, that by supplying gradient of reduced cost functional f, we have managed to significantly reduce the CPU-time in the semi-linear problem.

5.2.4 Projected gradient method

In this section we introduce the fourth optimization method, which is called *Projected gradient method*. This method is the faster version of *Conditioned gradient method*, which is quite slow as it converges linearly [5]. Again, consider the linear problem $P1$, where we want to minimise the reduced cost functional 5.19

$$f(\vec{u}) = \frac{1}{2}\|\Lambda\vec{u} - y_\Omega\|^2_{L^2(\Omega)} + \frac{\lambda}{2}\|\sum_{i=1}^{k} u_i e_i\|^2_{L^2(\Omega)} \qquad \vec{u} \in \vec{U}_{ad,k} \tag{5.31}$$

The algorithm proceeds as follows:

S1 Solve the state system of $P1$.

S2 Calculate the associated adjoint state p by solving the adjoint equation 5.29

$$\begin{cases} -\Delta p = y - y_\Omega & \text{in } \Omega \\ p = 0 & \text{on } \Gamma. \end{cases} \tag{5.32}$$

S3 Calculate $\nabla f(\vec{u})$

$$\nabla f(\vec{u}) = \begin{bmatrix} (p, e_1)_{L^2(\Omega)} \\ \vdots \\ (p, e_k)_{L^2(\Omega)} \end{bmatrix} + \lambda D\vec{u} \qquad \vec{u} \in \mathbb{R}^k, \tag{5.33}$$

and take the negative gradient as a descent direction

$$\vec{v} = -\nabla f(\vec{u}) \tag{5.34}$$

S4 Choose the initial step size $s_n = s_0 = 0.1$ and check if it is true:

$$f(\vec{u}_n + s_n\vec{v}) < f(\vec{u}_n) \tag{5.35}$$

True: $\vec{u}_{n+1} = \vec{u}_n + s_n\vec{v}$; False: $s_n = \frac{s_n}{2}$

S5 If$|\nabla f(\vec{u})| \leq \varepsilon$ - stop, otherwise start with **S1**.

For $P2$, we use *Newton's method* introduced in Chapter 3 to linearise the equation. Now, let us consider the semi-linear optimal control problem $P2$. We have the algorithm as follows:

S1 Solve the state system of $P2$.

S2 Find the associated adjoint state p by solving the adjoint equation 5.30.

We repeat the rest of the steps as in the linear case above.

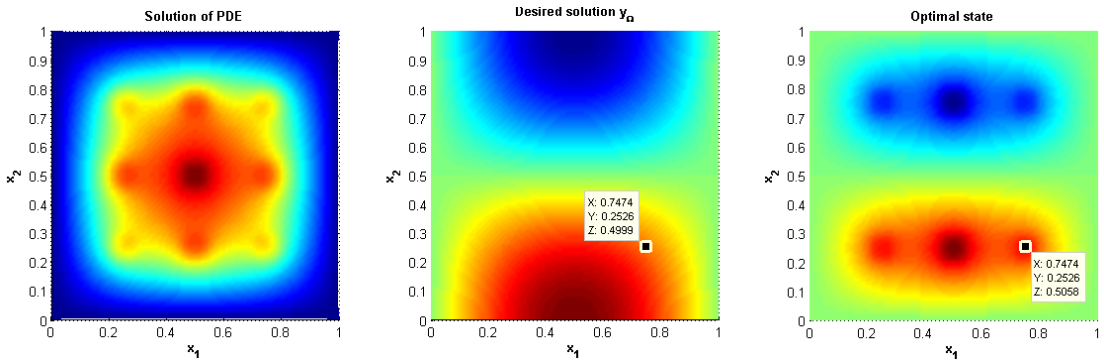


FIGURE 5.6: Optimization results of the optimal control problem $P2$ using FVM together with *Projected gradient method* code in 2D: $m = 96$, $k = 9$, initial control $u_i = 1$ and $E_i = 8 \times 8$ for $i = 1, \ldots, k$

Figure 5.6 also shows that the optimal solution at the E_i areas is almost identical to the desired solution.

Projected gradient method	**FVM**		**FEM**	
	Linear	Semi-linear	Linear	Semi-linear
CPU-time	0.987159	37.059861	2.227709	252.214029
$f(\vec{u})$	-0.0508	0.0717	-0.0507	0.0717

TABLE 5.4: Results of the *projected gradient method* optimization with supplied gradient of the problems $P1$ and $P2$ using FVM and FEM discretization

Table 5.4 shows the results of projected gradient method applied to the problems P1 and P2 together with FVM and FEM.

Now, let us take a look at the constraint example of P1 of quadprog method. We will consider the constraint problem P1, where constraints are as follows:

$$-200 \leq u_i \leq 200 \quad u_i \in U_{ad,k}, \tag{5.36}$$

Let us use quadprog with FVM. For the unconstrained example we get vector $\vec{u}$ as follows:

$$\vec{u} = \begin{bmatrix} 153.1022 \\ 218.6932 \\ 153.1022 \\ 0 \\ 0 \\ 0 \\ -153.1022 \\ -218.6932 \\ -153.1022 \end{bmatrix} \tag{5.37}$$

whereas for the constraint example it is:

$$\vec{u} = \begin{bmatrix} 155.4989 \\ 200.0000 \\ 155.4989 \\ -0.0000 \\ -0.0000 \\ -0.0000 \\ -155.4989 \\ -200.0000 \\ -155.4989 \end{bmatrix} \tag{5.38}$$

We can see how the constraints affects the control vector $\vec{u}$.

Chapter 6

Conclusion of the numerical results

In Chapter 5 we have found the minimization solution to the linear and semi-linear optimal control problems P1 and P2, respectively, by using different optimization techniques. First of all, we have transformed the problem into reduced quadratic optimization problem and used it to find the optimal solutions for the linear optimal control problem P1, while for the semi-linear problem P2 we have used the reduced cost functional. We have also applied four optimization techniques: quadprog, fmincon, fmincon with supplied gradient and projected gradient method. We have tested these methods using FVM and FEM.

In Chapter 6, will summarize all the results. First of all, the optimal state y of every Figure of P1 in Chapter 5 produced almost identical results. The same has happened for problem P2. This is a good indicator that all four optimization methods together with two different discretization techniques provide accurate results. This is well presented in Table 6.1:

P1 - linear	**FVM**		**FEM**	
	CPU-time	$f(\vec{u})$	CPU-time	$f(\vec{u})$
quadprog	1.57284	-0.0509	2.958231	-0.0508
fmincon	2.17389	-0.0509	3.553294	-0.0508
fmincon+∇	1.896378	-0.0505	3.42329	-0.0505
PGM	0.987159	-0.0508	2.227709	-0.0507

TABLE 6.1: Results of all four optimization techniques applied to the linear optimal control problem P1 using FVM and FEM discretization

From Table 6.1 we can see that all three methods produce very similar results for the linear problem P1. Overall, the computational time is very small. Even when the gradient is supplied to fmincon, the speed barely changes. However, the *Projected Gradient Method* (PGM) is shown to be the fastest.

P2 - semi-linear	**FVM**		**FEM**	
	CPU-time	$f(\vec{u})$	CPU-time	$f(\vec{u})$
fmincon	299.523994	0.0717	945.401196	0.0717
fmincon+∇	50.087365	0.0717	516.93376	0.0717
PGM	37.059861	0.0717	252.214029	0.0717

TABLE 6.2: Results of three optimization techniques applied to the semi-linear optimal control problem P2 using FVM and FEM discretization

From Table 6.2 we can see that all three methods produce very similar results, although the computational time has seriously increased in the case of semi-linear PDE in P2. It is clear that here fmincon with supplied gradient is a much better choice for optimization than the fmincon on its own. However, PGM is again the fastest method of them all.

Bibliography

[1] Jordan D. W. and Smith P. *Nonlinear Ordinary Differential Equations : An Introduction to Dynamical Systems.* Oxford University Press, Oxford, England, 1999.

[2] Van Iwaarden J. L. *Ordinary Differential Equations with Numerical Techniques.* Harcourt Brace Jovanovich, San Diego, 1985.

[3] Walter W. *Ordinary Differential Equations.* Springer, London, 1998.

[4] Vermolen F. van Kan J., Segal A. *Numerical Methods in Scientific Computing.* VSSD, Delft, 2008.

[5] Tröltzsch F. *Optimal Control of Partial Differential Equations*, volume 112. American Mathematical Society, 2010.

[6] Celia M. A. and Gray W. G. *Numerical Methods for Differential Equations : Fundamental Concepts for Scientific and Engineering Applications.* Prentice-Hall, Englewood Cliffs, NJ, 1992.

[7] Kevorkian J. *Partial Differential Equations : Analytical Solution Techniques.* Springer, New York, second edition, 2011.

[8] Bensoussan A. *Regularity Results for Nonlinear Elliptic Systems and Applications.* Springer, Berlin, 2001.

[9] Jost J. *Partial Differential Equations.* Springer, London, 2002.

[10] Smith G. D. *Numerical Solution of Partial Differential Equations: Finite Difference Methods.* Clarendon Press, Oxford, third edition, 1985.

[11] Lebedev L. P. and Cloud M. J. *The Calculus of Variations and Functional Analysis: With Optimal Control and Applications in Mechanics.* World Scientific, New Jersey, 2003.

[12] Lyashko S. I. *Generalized Optimal Control of Linear Systems with Distributed Parameters.* Kluwer Academic Publishers, New York, 2002.

[13] Hariharan S. I. and Moulden T. H. *Numerical Methods for Partial Differential Equations.* Longman, Harlow, Essex, 1986.

[14] Ames W. F. *Numerical Methods for Partial Differential Equations.* Academic Press, London, third edition, 1992.

[15] Ganzha V. G. and Vorozhtsov E. V. *Numerical Solutions for Partial Differential Equations: Problem Solving Using Mathematica.* CRC Press, New York, 1996.

[16] Cooper J. *Introduction to Partial Differential Equations with MATLAB.* Birkhäuser, Boston, 1998.

[17] Mathews J. H. *Numerical Methods Using MATLAB.* Prentice Hall, London, third edition, 1999.