

Finite Element Methods for Advection Problems in Hilbert Spaces

U7PSCFCP (G900)

MSc Dissertation in
Scientific Computation

2019/2020

School of Mathematical Sciences

University of Nottingham

Adam Howlett

Supervisor: Kristoffer Van der Zee

I have read and understood the School and University guidelines on plagiarism. I confirm that this work is my own, apart from the acknowledged references.

Abstract

This dissertation demonstrates the issues with classic Finite Element Schemes for one and two-dimensional advection problems in a Hilbert space context, and discusses the effectiveness of other approaches. The design of algorithms in terms of interacting objects allowed an application of the recently proposed technique of Discrete-Dual Residual Minimisation (DDMRes) with scope for generalisation of the system parameters. In particular, new test-spaces on the Peterson Mesh are proposed and investigated, with novel calculations performed in two-dimensional advection with constant flow on the unit square. This yielded optimal convergence rates in the L^2 -norm of the absolute error, as well as the norm of the residual in a defined graph space. Additionally, a specialised refinement of the Peterson mesh was found that afforded faster computation in a specific constant vertical flow setting, with the cost of an unsolvable system in others and a singular system in the horizontal flow case.

Contents

1	Introduction	4
2	Problem Overview	6
3	One Dimension: A Basic PDE	9
3.1	Standard Galerkin Formulation	10
3.2	A Least-Squares Approach	15
3.3	Closing Remarks on One-Dimensional Finite Element Methods	20
4	Two Dimensions: Advection-Reaction	21
4.1	Problem Overview	21
4.2	Discretisation	23
4.3	Implementation	26
4.4	Computations and Analysis	33
5	Conclusions	42
A	Appendix	44
A.1	Coercivity and Positive Definiteness	44
A.2	Generating Céa's Lemma from Well-Posedness Assumptions	45
A.3	Constructing the Standard Galerkin System in One Dimension	46
A.4	A Proof of the Singularity of (3.8) in the Odd Dimensional Case	47
A.5	The Equivalence of Problems (3.10) and (3.13)	48
A.6	Lax Milgram Criteria Verification for (3.10)	49
A.7	A Proof of the Discrete Inf-Sup Condition for (3.17)	50
A.8	Constructing the Least-Squares System in One Dimension	51
A.9	The Equivalence of Problems (4.11) and (4.12)	52
A.10	The Sufficiency of One Application of Red Refinement for DDMRes on the Peterson Mesh	54
A.11	Reduction to Petrov-Galerkin for the Vertical Refinement Scheme	56
B	References	57

1 Introduction

Finite element methods encapsulate some of the most powerful numerical tools for solving differential equations. They have important applications in modelling, engineering and more abstract mathematics where simple, analytic solutions to general problems are not possible or practical to obtain. As such, the inability of ‘standard’ methods to adequately solve non-coercive weak formulations of even simple first-order partial differential equations is a pressing issue. Procedural developments in the field of finite element methods have been rectifying this issue, with the introduction of new methods like discontinuous Petrov Galerkin schemes. In particular, the advection-reaction problem has proven notoriously difficult to solve and, as a result, there has been considerable academic focus its numerical solution.

This dissertation outlines specifically the impracticalities of such previous formulations, such as the ‘standard Galerkin’ methodology, and demonstrates formulations in both one and two dimensions that employ more sophisticated solution routines. Principles used in this dissertation, like the Discrete-Dual Minimal-Residual method (DDMRes), have been applied before to more general Banach space settings ([1]), but we will restrict ourselves to a Hilbert space context, with a focus towards the specific results for advection and efficiency of the developed routines. The dissertation investigates the effectiveness of both the least-squares formulation for one-dimensional advection, and DDMRes with a particular mesh (and refinement on that mesh) on the unit square for two-dimensional advection.

The least-squares method for advection in one dimension with Dirichlet boundary yields $\mathcal{O}(h^2)$ convergence to the solution in the L^2 -norm as long as the solution $u \in H^2(\Omega)$. Otherwise, sub-optimal convergence is observed. In constant flow scenarios, the DDMRes method demonstrated $\mathcal{O}(h)$ convergence in the L^2 -norm of the absolute error in the measurement, as well as the norm of the residual in a relevant graph space. The Peterson Mesh itself was built to demonstrate ‘cross-diffusion’ in certain methods, which implies sub-optimal convergence and is known to converge only in $\mathcal{O}(h^{\frac{1}{2}})$ with the implementation of Discontinuous Galerkin methods ([10]). As a result, an optimal convergence rate

with DDMRes for any test case in this setting is not trivial. Additionally, in the vertical flow case, a specific refinement of the mesh was constructed that resulted in the same $\mathcal{O}(h)$ convergence but was more computationally efficient, as a result of the lack of the additional unnecessary computation of integrals, and a smaller-sized system of equations to solve for. However, the same mesh demonstrated a singular system in some cases, and due to its ‘purpose-built’ construction can not be used to solve DDMRes on the Peterson Mesh excepting very specific cases.

The remainder of this dissertation will continue as follows: In Section 2, we will give an overview of the first-order problem starting from first-principles, and give a dimensionally independent statement of the issue with non-coercive formulations. In Section 3, we will focus on one-dimensional advection and the failure of the standard Galerkin formulation to solve it, turning to the least-squares method to demonstrate its effectiveness. In Section 4, we will walk through two-dimensional advection in terms of its weak form, well-posedness, and the specific discretisation we will be using. After detailing the routines developed to set up and solve the system, we will investigate its effectiveness at solving advection problems for several test-cases. We will conclude these results in Section 5, reflecting on them and discussing further options for study for this setting.

2 Problem Overview

A general first-order PDE can be considered in the abstract form:

Find $u \in \mathbb{U}$ s.t.

$$Au = f \quad \text{in } \mathbb{V}^* \quad (2.1)$$

where \mathbb{U}, \mathbb{V} are Banach spaces, $A : \mathbb{U} \rightarrow \mathbb{V}^*$ is a linear operator and f is some ‘source’ data (or more precisely, some given element in the dual space \mathbb{V}^* of \mathbb{V}). The problem is converted into a ‘weak formulation’, given as:

Find $u \in \mathbb{U}$ s.t.

$$\langle Au, v \rangle_{\mathbb{V}^*, \mathbb{V}} = \langle f, v \rangle_{\mathbb{V}^*, \mathbb{V}} \quad \forall v \in \mathbb{V} \quad (2.2)$$

where $\langle \cdot, \cdot \rangle$ is some duality pairing. In our setting, we will write this problem in the following form:

Find $u \in \mathbb{U}$ s.t.

$$b(u, v) = l(v) \quad \forall v \in \mathbb{V}. \quad (2.3)$$

Here, $b(\cdot, \cdot)$ is a bilinear form on $\mathbb{U} \times \mathbb{V}$, and $l(\cdot)$ is a linear form on \mathbb{V} , which is to say

$$a(\alpha u + \beta w, v) = \alpha a(u, v) + \beta a(w, v) \quad \forall \alpha, \beta \in \mathbb{R}, u, w \in \mathbb{U}, v \in \mathbb{V}, \quad (2.4a)$$

$$a(u, \alpha v + \beta w) = \alpha a(u, v) + \beta a(u, w) \quad \forall \alpha, \beta \in \mathbb{R}, u \in \mathbb{U}, v, w \in \mathbb{V}, \quad (2.4b)$$

$$l(\alpha v + \beta w) = \alpha l(v) + \beta l(w) \quad \forall \alpha, \beta \in \mathbb{R}, v \in \mathbb{V}. \quad (2.4c)$$

When $\mathbb{U} = \mathbb{V}$, the existence and uniqueness of a solution to the weak form as stated in (2.3) is guaranteed by the satisfaction of the Lax-Milgram criteria, although not exclusively. Specifically, it requires that the bilinear form $b(\cdot, \cdot)$ be:

- Coercive: i.e $\exists c_0 \in \mathbb{R} > 0$ s.t. $\forall v \in \mathbb{V}, b(v, v) \geq c_0 \|v\|_{\mathbb{V}}^2$,
- Continuous: i.e $\exists c_1 \in \mathbb{R} > 0$ s.t. $\forall w, v \in \mathbb{U}, \mathbb{V}, |b(w, v)| \leq c_1 \|w\|_{\mathbb{U}} \|v\|_{\mathbb{V}}$,

and additionally requires that $l(\cdot)$ be:

- Continuous: i.e $\exists c_2 \in \mathbb{R} > 0$ s.t. $\forall v \in \mathbb{V}, |l(v)| \leq c_2 \|v\|_{\mathbb{V}}$

Performing a finite element method involves discretising the problem (2.3) by taking $\mathbb{U} \rightarrow \mathbb{U}_h$, some finite-dimensional subspace of \mathbb{U} (defined by a ‘mesh-parameter’ h), and $\mathbb{V} \rightarrow \mathbb{V}_h$, similarly. We may now write our problem:

Find $u_h \in \mathbb{U}_h$ s.t.

$$b(u_h, v_h) = l(v_h) \quad \forall v_h \in \mathbb{V}_h. \quad (2.5)$$

In a standard Galerkin setting, $\mathbb{U}_h = \mathbb{V}_h$, and we have some basis $\phi_1, \phi_2, \dots, \phi_n$ for \mathbb{U}_h . We can represent our solution u_h as a linear combination of these basis functions:

$$u_h = \sum_{i=1}^n U_i \phi_i \quad (2.6)$$

where the coefficients U_i will define the solution. If we then insert this into (2.5) along with an analogous expansion for v_h , we will find

$$\begin{aligned} b\left(\sum_{i=1}^n U_i \phi_i, \sum_{j=1}^n V_j \phi_j\right) &= l\left(\sum_{j=1}^n V_j \phi_j\right), \\ \rightarrow \sum_{i=1}^n U_i \left(\sum_{j=1}^n V_j (a(\phi_i, \phi_j))\right) &= \sum_{j=1}^n V_j (l(\phi_j)), \quad (\text{by bilinearity}) \\ \rightarrow \sum_{i=1}^n U_i (a(\phi_i, \phi_j)) &= l(\phi_j). \end{aligned}$$

This is now equivalent to a linear system of equations $\mathbf{A}\mathbf{U} = \mathbf{F}$, where

$$A_{ij} = a(\phi_i, \phi_j), \quad F_i = l(\phi_i), \quad (2.7)$$

and $\mathbf{U} = (U_1, U_2, \dots, U_n)^T$ is the set of coefficients that defines the solution. From the coercivity of $b(\cdot, \cdot)$, we know that the matrix \mathbf{A} is positive definite (see A.1 for a proof of this). This guarantees its invertibility, and thus the existence of a unique solution for any data set \mathbf{F} . Additionally, as a direct result of the Lax-Milgram criteria, we have the following error bound:

$$\|u - u_h\|_{\mathbb{U}} \leq \frac{c_1}{c_0} \inf_{v_h \in \mathbb{U}_h} \|u - v_h\|_{\mathbb{U}}. \quad (2.8)$$

This is a statement of the stability of the problem in the sense that it is always bounded by the best possible approximation in the norm of \mathbb{U} , up to some constant defined by the coercivity and continuity parameters.

We see now, how in this standard Galerkin setting, the guarantee of a well-posed problem with a discretisation solution that converges in the relevant norm is contingent on

the Lax-Milgram criteria. Since many such variational forms of first-order PDEs are non-coercive, we have a serious issue. As a result of this, we must expand our understanding of the criteria for well-posedness to include those Hilbert space settings where $\mathbb{U} \neq \mathbb{V}$ in general. We look to the Banach-Nečas-Babuška Theorem (BNB), of which the Lax-Milgram Theorem is a special case. We now require the conditions:

$$\exists \gamma > 0 \text{ s.t. } \inf_{w \in \mathbb{U}} \sup_{v \in \mathbb{V}} \frac{b(w, v)}{\|w\|_{\mathbb{U}} \|v\|_{\mathbb{V}}} \geq \gamma, \quad (2.9a)$$

$$\{v \in \mathbb{V} : b(w, v) = 0 \ \forall w \in \mathbb{U}\} = \{0\}. \quad (2.9b)$$

Together with continuity, these two statements are equivalent to the well-posedness of a given problem of the form (2.3) (see e.g. [2], Thm. 2.4). Similarly, the discretised problem is well-posed if it satisfies the ‘discrete inf-sup condition’ ([2], Thm. 2.8):

$$\exists \hat{\gamma} > 0 \text{ s.t. } \inf_{w_h \in \mathbb{U}_h} \sup_{v_h \in \mathbb{V}_h} \frac{b(w_h, v_h)}{\|w_h\|_{\mathbb{U}} \|v_h\|_{\mathbb{V}}} \geq \hat{\gamma}, \quad (2.10)$$

of which Céa’s Lemma is a natural result (see (A.2)):

$$\|u - u_h\|_{\mathbb{U}} \leq \left(1 + \frac{c_1}{\hat{\gamma}}\right) \inf_{w_h \in \mathbb{U}_h} \|u - w_h\|_{\mathbb{U}} \quad (2.11)$$

We note here that, if we have coercivity, BNB is equivalent to Lax-Milgram and Céa’s lemma is equivalent to (2.8) ([3] - Thm 7, [4] Lemma 2.28). So we have some exclusive condition for well-posedness, but we need to specify how a solution is going to behave as our mesh parameter h changes. In particular, we would like ‘approximability’, defined as:

$$\lim_{h \rightarrow 0} \left(\inf_{w_h \in \mathbb{U}} \|u - w_h\|_{\mathbb{U}} \right) = 0 \quad \forall u \in \mathbb{U}. \quad (2.12)$$

If we have approximability, it follows from (2.11) that not only does a unique solution to (2.5) exist, but it will converge to the correct solution as $h \rightarrow 0$. Typically, we may make use of standard results for polynomial interpolation to show approximability, but we will additionally verify convergence, usually in terms of $\mathcal{O}(h^\alpha)$, in an experimental sense.

We will look now at a simple first-order PDE in one dimension, demonstrate the issues with a standard Galerkin discretisation, and discuss another method to generate more stable solutions. This will serve as a smaller representation of our approach to solving other systems (or specifically the advection-reaction equation) where more complexity is involved.

3 One Dimension: A Basic PDE

Our simple, one-dimensional PDE will be of the form:

$$u_x = f(x) \quad \forall x \in \Omega = (0, 1) \quad (3.1a)$$

$$u(0) = 0. \quad (3.1b)$$

We wish to find $u(x)$ given some ‘source’ data, $f(x)$. A single boundary condition is enough to specify an exact solution for this problem. Generally, our weak formulation will be generated by applying some test function $v(x) \in \mathbb{V}$ and integrating over the domain to obtain the problem:

Find $u \in \mathbb{U}$ s.t.

$$\int_{\Omega} u(x)v(x)d\Omega = \int_{\Omega} f(x)v(x)d\Omega \quad \forall v \in \mathbb{V}. \quad (3.2)$$

We will see over the course of this section that the methods involved will specify \mathbb{U} , \mathbb{V} , and $v(x)$ in slightly different ways, and that the well-posedness of the problem and the convergence of any possible solution will be ultimately dependent on this, alongside the particular method of discretisation performed.

3.1 Standard Galerkin Formulation

3.1.1 Problem Definition

We begin by defining the space

$$H_0^1(\Omega) = \{w(x) \in L_2(\Omega); w'(x) \in L_2(\Omega); w(0) = 0\} \quad (3.3)$$

whose norm is the H^1 -norm:

$$\|w\|_{H^1(\Omega)} = (\|w'\|_{L_2}^2 + \|w\|_{L_2}^2)^{\frac{1}{2}} \quad (3.4)$$

where the derivative $w'(x)$ is defined in the weak sense. A key feature of the standard Galerkin method is the equality of the spaces \mathbb{U} and \mathbb{V} . We note $u(x) \in H_0^1(\Omega)$ and thus represent (3.1) in weak form by multiplying by some test function $v(x) \in H_0^1(\Omega)$ to restate the problem:

Find $u \in H_0^1(\Omega) \subset L_2(\Omega)$ s.t.

$$\int_{\Omega} u'(x)v(x)dx = \int_{\Omega} f(x)v(x)dx \quad \forall v \in H_0^1(\Omega), \quad (3.5)$$

or alternatively:

Find $u \in H_0^1(\Omega) \subset L_2(\Omega)$ s.t.

$$b(u, v) = l(v) \quad \forall v \in H_0^1(\Omega). \quad (3.6)$$

3.1.2 Well-Posedness

Verification of the Lax-Milgram Criteria is simple in this case, excepting coercivity, for which we find

$$\begin{aligned} b(w, w) &= \int_{\Omega} w'w dx \\ &= \frac{1}{2} \int_{\Omega} (w^2)' dx \\ &= \frac{1}{2} [w^2]_0^1 \\ &= \frac{1}{2} w^2(1). \end{aligned}$$

Since we require some positive constant c_0 s.t. $b(w, w) \geq c_0 \|w\|_{H^1(\Omega)}^2 \quad \forall w \in H_0^1(\Omega)$, to disprove coercivity it suffices to find some counter-example $w(x)$ that vanishes on the boundaries, but has nonzero norm in $H^1(\Omega)$. $\sin(\pi x)$ for example, has this property.

When dealing with second-order PDEs, a common approach is to perform integration by parts on the weak form and restate the problem in terms of the resultant integral and boundary terms. As coercivity is a property of the specific bilinear form in question, rewriting the equation in this way may allow it to be satisfied. However, It quickly becomes apparent that this does not solve the issue:

$$\begin{aligned}
b(u, v) &= \int_{\Omega} u'(x)v(x)dx \\
&= - \int_{\Omega} u(x)v'(x)dx + [u(x)v(x)]_0^1 \\
&= - \int_{\Omega} u(x)v'(x)dx + u(1)v(1) \quad (u(0) = 0).
\end{aligned}$$

We can see that the form of $b(w, w)$ will be identical, save for the additional term in $w(1)$, which would vanish by letting $w(1) = 0$ in the same way as before. This does not necessarily mean our problem is ill-posed, just that we have not yet been able to guarantee a unique solution. On discretisation, we will turn to the discrete inf-sup condition to help reconcile this.

3.1.3 Discretisation

We discretise, taking $H_0^1(\Omega) \rightarrow \mathbb{U}_h = \mathbb{P}_{\text{cont}}^1(\mathcal{T}_n)$, where \mathcal{T}_n is the subdivision of Ω into a uniform set of elements $[x_i, x_{i+1}]$, for $i = 0, 1, \dots, n-1$ where $x_i = ih$ and $h = \frac{1}{n}$. Our basis for \mathbb{U}_h will be a set of ‘hat functions’ defined on these elements:

$$\phi_i(x) = \left(1 - \left|\frac{x - x_i}{h}\right|\right)_+ \quad i = 1, \dots, n-1. \quad (3.7)$$

We should note that this set of basis functions implicitly sets $u_h(1) = 0$ (due to the lack of a basis function ϕ_n). As a result, we must be wary to choose test cases with f that also imply $u(1) = 0$ when the boundary condition of (3.1) is applied. Otherwise, our results will likely not make sense. For a construction of this form, it is trivial to show (see A.3) that our $(n-1) \times (n-1)$ system of equations, $\mathbf{A}\mathbf{U} = \mathbf{F}$, is given by

$$\begin{pmatrix}
0 & \frac{1}{2} & \cdots & 0 & 0 \\
-\frac{1}{2} & 0 & \cdots & & 0 \\
\vdots & & \ddots & & \vdots \\
0 & & \cdots & 0 & \frac{1}{2} \\
0 & 0 & \cdots & -\frac{1}{2} & 0
\end{pmatrix}
\begin{pmatrix}
U_1 \\
U_2 \\
\vdots \\
U_{n-2} \\
U_{n-1}
\end{pmatrix}
=
\begin{pmatrix}
F_1 \\
F_2 \\
\vdots \\
F_{n-2} \\
F_{n-1}
\end{pmatrix}. \quad (3.8)$$

We could, at this point, attempt to verify mathematically that the discrete inf-sup condition holds (or does not hold) for this formulation. However, since it states that all matrices \mathbf{A} will have a unique solution for any \mathbf{F} (and therefore be invertible), a singular counter-example is equivalent to disproving it. It is not difficult to show that, for odd degrees of freedom $(n - 1)$, the matrix \mathbf{A} is singular (see (A.4) for a proof), and thus we do not have a unique solution. As a result, there is no inf-sup constant that applies to the entire family of systems defined on \mathbb{U}_h , and we have no guarantee of well-posedness in the even case.

3.1.4 Computations and Analysis

In this discretisation, the even-numbered $(n - 1)$ cases are invertible, affording solutions to the problem in these spaces. However, these solutions develop oscillations in the approximation, even for simple functions. For example, if we take $u(x) = \sin(\pi x) \rightarrow f(x) = \pi \cos(\pi x)$, we can generate the approximations for decreasing h seen in Figure 1.

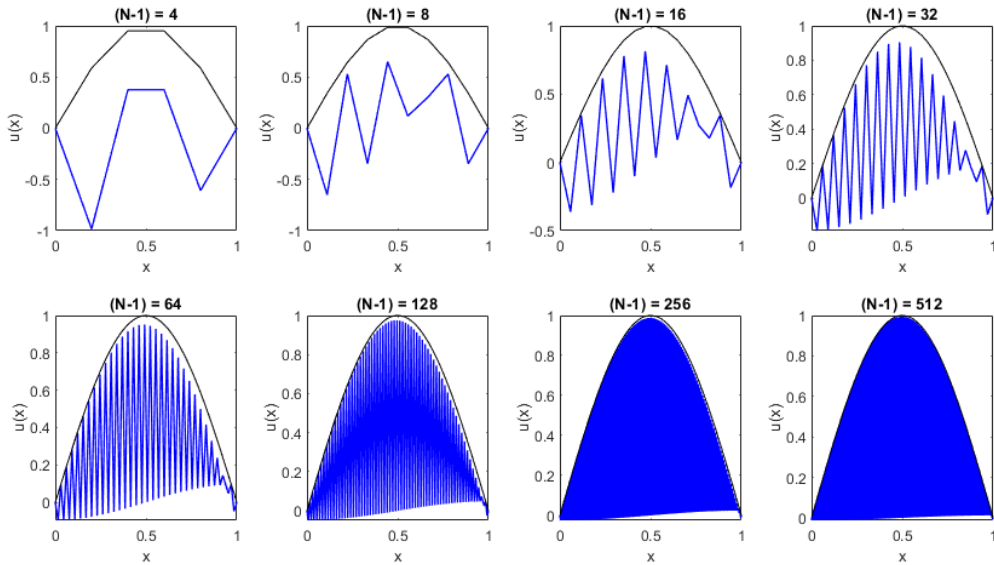


Figure 1: The $u_h \sim \sin(\pi x)$ approximation with the standard Galerkin method for increasing system size.

It appears that the approximation at least ‘converges’ to the correct solution $u(x)$ in some nebulous sense, but while oscillating backwards and forwards between a reasonable approximation and some other seemingly arbitrary values. If we plot instead a linear interpolation of every odd-numbered coefficient, we would find something that looks a lot more like an approximation to $\sin(\pi x)$ (see Figure 2).

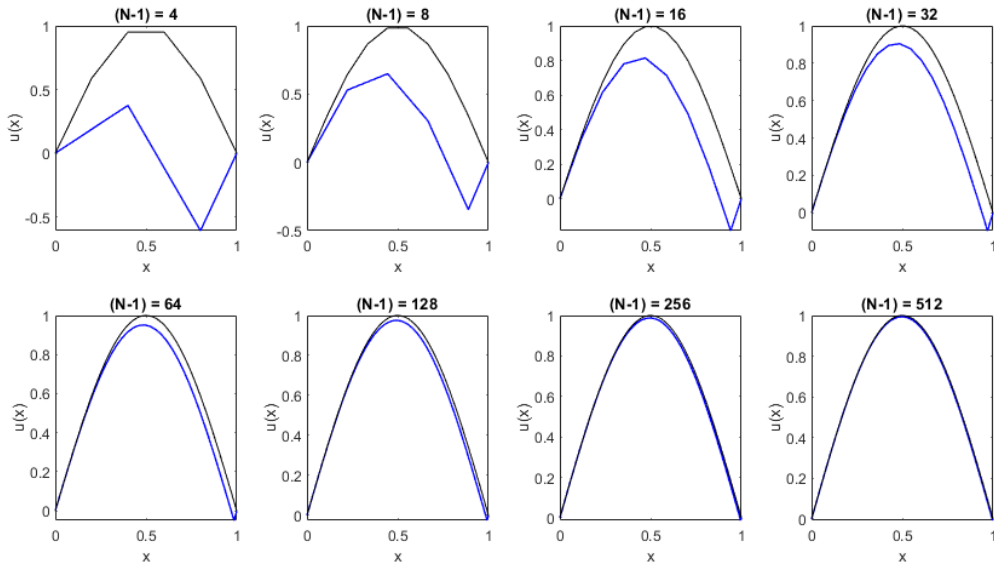


Figure 2: Interpolation of the odd coefficients of the $u_h \sim \sin(\pi x)$ approximation with the standard Galerkin method for increasing system size.

However, this is a relatively arbitrary construction, and although we could feasibly demonstrate convergence in the H_1 -norm for this specific example, the fundamental problem with this discretisation is its lack of discrete inf-sup conformity in general. We could not in any sense expect to find a similar convergence for arbitrary data, and could expect an unstable or perhaps even divergent solution.

Examining the determinants of our systems with even-numbered degrees of freedom, we observe the following, verified computationally for at least $(n - 1) \leq 10$:

$$\det(\mathbf{A}) = 2^{-(n-1)}$$

so here we have $\det(\mathbf{A}) \rightarrow 0$ as $h \rightarrow 0$. This may provide some statement about an approaching ‘singularity’ of the matrix in some sense, or some associated instability, but without a direct reference to the discrete-inf-sup condition here we could not precisely say.

If we are to add an additional basis function to this construction, the solutions that are generated do not demonstrate an much of an improvement. A full treatment of this discretisation is available in [4], Sec. 5.5, in which Ern and Guermond demonstrate that the discrete inf-sup constants exist, but are h -dependent. Thus, as $h \rightarrow 0$, the solution becomes rapidly unstable. For a general source, the standard Galerkin formulation will ultimately fail in its attempt to approximate the solution to (3.1).

3.2 A Least-Squares Approach

3.2.1 Problem Definition

The least-squares approach attempts to solve problem (3.1) by first defining an analogous problem to (3.2). Consider the functionals:

$$\tilde{b}(w, v) = \int_{\Omega} w'(x)v'(x)d\Omega \quad \forall w, v \in H^1(\Omega) \quad (3.9a)$$

$$\tilde{l}(v) = \int_{\Omega} f(x)v'(x)d\Omega \quad \forall v \in H^1(\Omega) \quad (3.9b)$$

and the new problem:

Find $u \in H_{(0)}^1(\Omega)$ s.t.

$$\tilde{b}(u, v) = \tilde{l}(v) \quad \forall v \in H_{(0)}^1(\Omega). \quad (3.10)$$

Notice here we are setting $\mathbb{U} = \mathbb{V}$, which is not necessarily the case for (3.2). Before we focus on solving this, we may want to note that if we choose the spaces for (3.2) and rewrite it as

Find $u \in H_{(0)}^1(\Omega)$ s.t.

$$b(u, v) = l(v) \quad \forall v \in L^2(\Omega) \quad (3.11)$$

then our problems are actually equivalent: u solves (3.10) iff u solves (3.11) (see [4], Prop. 5.17). Furthermore, if we define the functional

$$J(w) = \frac{1}{2}\tilde{b}(w, w) - \tilde{l}(w) = \frac{1}{2} \int_{\Omega} (w')^2 d\Omega - \int_{\Omega} f w' d\Omega, \quad (3.12)$$

we may note that, since $\tilde{b}(u, v)$ is symmetric, (3.10) is also equivalent to the minimisation problem:

Find $u \in H_{(0)}^1$ s.t

$$u = \arg \min_{w \in \mathbb{H}_{(0)}^1} J(w). \quad (3.13)$$

This is where the least-squares method gets its name (see A.5 for a proof), and the link between weak formulations of partial differential equations and minimisation problems remains an important observation during any study of their numerical solution.

3.2.2 Well-Posedness

Given the equivalence of problems (3.10) and (3.11), (3.10) may ‘inherit’ its well-posedness from a proof applied to (3.11). However, we can show that $\tilde{b}(\cdot, \cdot)$ is actually a coercive form:

$$\begin{aligned}
\tilde{b}(v, v) &= \int_0^1 |v'|^2 dx = \|v'\|_{L^2(\Omega)}^2 \\
&\geq 2\|v\|_{L^2(\Omega)}^2 && \text{(by the Poincaré-Friedrichs Inequality)} \\
\rightarrow \frac{1}{2}\tilde{b}(v, v) &= \frac{1}{2}\|v'\|_{L^2(\Omega)}^2 \\
&\geq \|v\|_{L^2(\Omega)}^2 \\
\rightarrow \frac{3}{2}\tilde{b}(v, v) &= \frac{3}{2}\|v'\|_{L^2(\Omega)}^2 \\
&\geq \|v\|_{L^2(\Omega)}^2 + \|v'\|_{L^2(\Omega)}^2 \\
\rightarrow \tilde{b}(v, v) &\geq \frac{2}{3}\|v\|_{H^1(\Omega)}^2.
\end{aligned} \tag{3.14}$$

Moreover, if we examine $\tilde{b}(\cdot, \cdot)$ we notice it is in the form attained by performing integration by parts on the bilinear form associated with the weak form of the classic second order model problem:

Find $u(x)$ s.t.

$$u''(x) = f(x). \tag{3.15}$$

on which the Lax-Milgram criteria are known to hold, given appropriate boundary conditions (see A.6). Thus, we can safely assume well-posedness and expect any solution to be bounded in error according to (2.8).

3.2.3 Discretisation

Similarly to the Galerkin formulation, we will discretise by taking $H_{(0)}^1(\Omega) \rightarrow \mathbb{U}_h = \mathbb{V}_h = \mathbb{P}_{\text{cont}}^1(\mathcal{T}_n)$ where \mathcal{T}_n is again the subdivision of Ω into elements $[x_i, x_{i+1}]$, for $i = 0, 1, \dots, n-1$ where $x_i = ih$ and $h = \frac{1}{n}$. Our basis for \mathbb{U}_h are again the hat functions (3.7), but with the addition of a final ‘half-hat’ function ϕ_n :

$$\phi_n = \begin{cases} 1 - \left| \frac{x-1}{h} \right| & 1-h \leq x \leq 1, \\ 0 & \text{otherwise.} \end{cases} \tag{3.16}$$

In this setting, we find that the discrete inf-sup condition holds and we are thus expecting an invertible matrix and a unique solution (see A.7 for a proof of this). Knowing this, we set up our system as before (see A.8), now of size $n \times n$, and find:

$$\frac{1}{h} \begin{pmatrix} 2 & -1 & \cdots & 0 & 0 \\ -1 & 2 & \cdots & & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & & \cdots & 2 & -1 \\ 0 & 0 & \cdots & -1 & 1 \end{pmatrix} \begin{pmatrix} U_1 \\ U_2 \\ \vdots \\ U_{n-2} \\ U_{n-1} \end{pmatrix} = \begin{pmatrix} F_1 \\ F_2 \\ \vdots \\ F_{n-2} \\ F_{n-1} \end{pmatrix}, \quad (3.17a)$$

with

$$F_i = \int_0^1 f(x) \phi'_i(x) dx = \frac{1}{h} \left(\int_{x_{i-1}}^{x_i} f(x) dx - \int_{x_i}^{x_{i+1}} f(x) dx \right). \quad (3.17b)$$

Before we continue, we might remark that the system we would get from a Petrov-Galerkin formulation of (3.11) would be functionally identical. This follows from the idea that the ‘natural’ discretisation space for $H^1(\Omega)$ is a piecewise linear space $\mathbb{P}_h^1(\Omega)$, while for $L^2(\Omega)$ we would choose a piecewise constant space $\mathbb{P}_h^0(\Omega)$. Since the derivative operator maps $\mathbb{P}_h^1(\Omega)$ to $\mathbb{P}_h^0(\Omega)$, the problem statements at the discrete level are also the same. Since we are approximating in a well-posed environment with piecewise linear interpolation, we may note that convergence is implied by the following standard result (see [5], Prop. 1.2):

$$\|u - u_h\|_{L^2(\Omega)} \leq Ch^2 \|u''\|_{L^2}. \quad (3.18)$$

So we expect a convergence rate of $\mathcal{O}(h^2)$, or a reduction of a factor four of the error in the L^2 -norm as a result of halving h . This, of course, requires at least $u \in H^2(\Omega)$. We will present test cases for both $u \in H^2(\Omega)$ and $u \notin H^2(\Omega)$ to demonstrate how this affects convergence.

3.2.4 Computations and Analysis

Solutions for these functions converge graphically very quickly, and by eye the graphs of the solution and its approximation are indistinguishable. For example, take the following approximations of the function $u(x) = 10x(1-x)(x - \frac{1}{4})$:

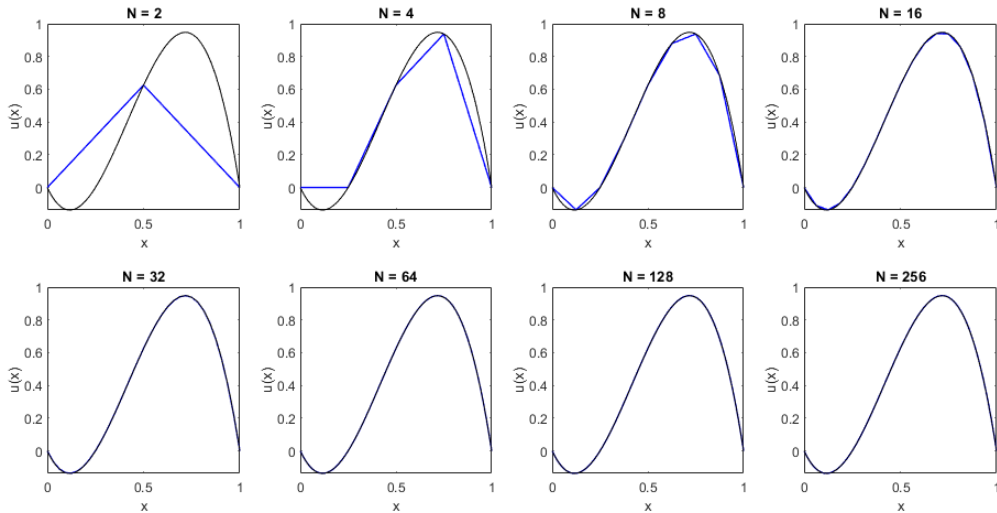


Figure 3: The $u_h \sim 10x(1-x)(x - \frac{1}{4})$ approximation with the least-squares method for increasing system size.

This demonstrates some significant approximability on its own, and a set of log-log plots of the error in the L^2 -norm against the system size n for various functions (Figure 4) shows $\mathcal{O}(h^2)$ convergence (so long as $u(x) \in H^2(\Omega)$). The first four plots here are for arbitrary functions with $u \in H^2(\Omega)$ (in fact, most if not all are in $H^\infty(\Omega)$). All four demonstrate $\mathcal{O}(h^2)$ convergence in the L^2 -norm, corresponding to a slope of ~ -2 . The last two plots have $u(x) \notin H^2(\Omega)$, and so their lack of second derivative means (3.18) does not apply. While they appear to be converging, they are doing so sub-optimally.

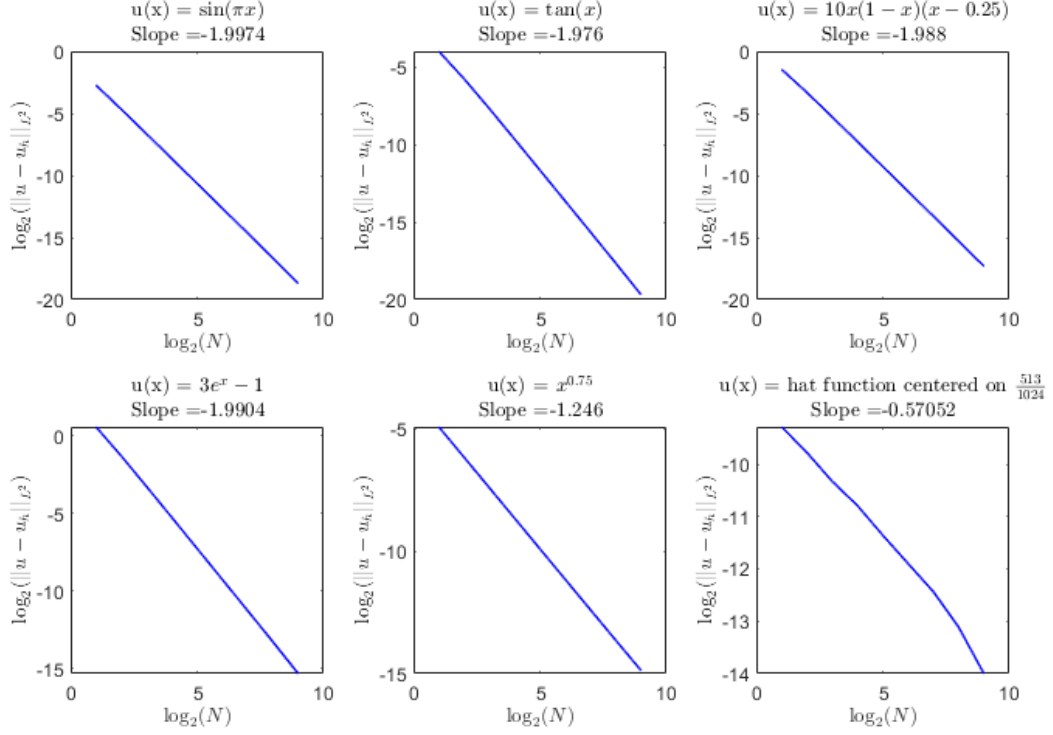


Figure 4: Log-Log plots for the errors associated with the approximations of several functions $u(x) \in H^2$ and $u(x) \notin H^2$ calculated with the least-squares method.

The least-squares method is appropriate for many functions in one dimension. Its key difference from the standard Galerkin method was a choice of the combination of spaces and problem setting such that the discrete inf-sup condition holds. If we have this, we will always have a well-posed problem.

3.3 Closing Remarks on One-Dimensional Finite Element Methods

3.3.1 Other Methods

For partial differential equations of first-order (or more generally, odd order), classic finite element methods do not give sufficient approximability. A whole range of additional methods in one dimension and beyond have been developed in part to deal with this issue, (each with their own particular advantages and disadvantages), including:

- Diffusion methods inc. Streamline Diffusion methods ([6], Sec. 9.7).
- Variational Multiscale methods [7].
- Discontinuous Galerkin and Petrov-Galerkin methods [8].
- Minimal-Residual methods inc. DDMRes [1].

In our shift to two dimensions, we will be focusing on the last of this list, one of the newer developments in finite element methods, and performing novel calculations on a set of meshes to demonstrate convergence of its approximation to the solution of the advection-reaction equation.

4 Two Dimensions: Advection-Reaction

4.1 Problem Overview

4.1.1 Equation, Variational Forms and Graph Spaces

The general form of the equation under consideration in two dimensions is:

$$\boldsymbol{\beta}(x) \cdot \nabla u + \mu(x)u = f(x) \quad u \in \Omega \subset \mathbb{R}^2, \quad (4.1a)$$

$$u = g(x) \quad u \in \Gamma_-, \quad (4.1b)$$

where $\boldsymbol{\beta} : x \rightarrow \mathbb{R}^2$ is some vector field over Ω (termed the *flow* or *velocity* field), $\mu : x \rightarrow \mathbb{R}$ is the ‘reaction rate coefficient’, f is our source data, and g is some nonhomogeneous boundary data (specified purely on the inflow). The boundary Γ is split into three parts - the *inflow*, *outflow*, and *characteristic* sections:

$$\Gamma_- = \{x \in \Gamma : \boldsymbol{\beta}(x) \cdot \hat{\mathbf{n}}(x) < 0\} \quad (4.2a)$$

$$\Gamma_+ = \{x \in \Gamma : \boldsymbol{\beta}(x) \cdot \hat{\mathbf{n}}(x) > 0\} \quad (4.2b)$$

$$\Gamma_0 = \{x \in \Gamma : \boldsymbol{\beta}(x) \cdot \hat{\mathbf{n}}(x) = 0\} \quad (4.2c)$$

where $\hat{\mathbf{n}}$ is the unit-outward normal on Γ . To construct our variational form, we will allow $\mu(x) = 0$ (focusing purely on Advection, the difficult part of the analysis) and integrate over both sides of the equation with a test function $v \in L^2(\Omega)$ to obtain

$$\int_{\Omega} (\boldsymbol{\beta} \cdot \nabla u) v \, d\Omega = \int_{\Omega} f v \, d\Omega. \quad (4.3)$$

By performing this step, we are implicitly requiring u to be a member of a *graph space* on Ω , during the construction of which we can impose the boundary conditions:

$$W_-(\boldsymbol{\beta}; \Omega) = \{v \in L^2(\Omega) : (\boldsymbol{\beta} \cdot \nabla v) \in L^2(\Omega), v = g \text{ on } \Gamma_-\}. \quad (4.4)$$

A different (more ‘weak’) form may be obtained by performing an integration by parts on the left hand side of the equation. We find:

$$\int_{\Omega} (\boldsymbol{\beta} \cdot \nabla u) v \, d\Omega = - \int_{\Omega} u (\nabla \cdot \boldsymbol{\beta} v) \, d\Omega + \int_{\Gamma} u v (\boldsymbol{\beta} \cdot \hat{\mathbf{n}}) d\Gamma.$$

The boundary term can be written as a combination of integrals over the three sections:

$$\begin{aligned}\int_{\Gamma} uv(\boldsymbol{\beta} \cdot \hat{\mathbf{n}}) &= \int_{\Gamma_+} uv(\boldsymbol{\beta} \cdot \hat{\mathbf{n}}) d\Gamma_+ + \int_{\Gamma_0} uv(\boldsymbol{\beta} \cdot \hat{\mathbf{n}}) d\Gamma_0 + \int_{\Gamma_-} uv(\boldsymbol{\beta} \cdot \hat{\mathbf{n}}) d\Gamma_- \\ &= \int_{\Gamma_+} uv(\boldsymbol{\beta} \cdot \hat{\mathbf{n}}) d\Gamma_+ + \int_{\Gamma_-} uv(\boldsymbol{\beta} \cdot \hat{\mathbf{n}}) d\Gamma_- \quad (\text{since } \boldsymbol{\beta} \cdot \hat{\mathbf{n}} = 0 \text{ on } \Gamma_0).\end{aligned}$$

If we impose on v membership of the graph space

$$W_+(\boldsymbol{\beta}; \Omega) = \{v \in L^2(\Omega) : \nabla \cdot (\boldsymbol{\beta}v) \in L^2(\Omega), v = 0 \text{ on } \Gamma_+\} \quad (4.5)$$

then we may write the weak form as

$$-\int_{\Omega} u(\nabla \cdot \boldsymbol{\beta}v) d\Omega = \int_{\Omega} fv d\Omega + \int_{\Gamma_-} gv|\boldsymbol{\beta} \cdot \hat{\mathbf{n}}| d\Gamma_- \quad (4.6)$$

since $u = g$, $\boldsymbol{\beta} \cdot \hat{\mathbf{n}} < 0$ on Γ_- .

4.1.2 Well-Posedness

The well-posedness of this equation in the Hilbert space setting is known in the case of non-homogeneous boundary conditions, subject to several restrictions on $g, \boldsymbol{\beta}$, and Γ ([9], Thm. 2.12). The most important of these for our purposes is the requirement that the inflow and outflow boundaries are ‘well-separated’ (i.e the smallest distance between an element of Γ_+ and Γ_- is nonzero). However, as noted by ([1], Remark 2.9 and Theorem A), if $\boldsymbol{\beta}$ is sufficiently regular, then we still have well-posedness by way of the assertion that the system is solvable with the method of characteristics. For the remainder of our analysis, we will assume $\boldsymbol{\beta}$ is constant, which satisfies this condition.

4.2 Discretisation

4.2.1 Petrov-Galerkin

In the one-dimensional case, we might wish to apply a Petrov-Galerkin methodology, where we take $\mathbb{U} \rightarrow \mathbb{P}^0(\mathcal{T}_n)$, $\mathbb{V} \rightarrow \mathbb{P}_{\text{cont}}^1(\mathcal{T}_n)$ and our problem statement (letting $\beta = 1, g = 0$) becomes:

Find $u_h \in \mathbb{P}^0(\mathcal{T}_n)$ s.t.

$$-\int_{\Omega} u_h v_h' d\Omega = \int_{\Omega} f v_h d\Omega \quad \forall v_h \in \mathbb{P}_{\text{cont}}^1(\mathcal{T}_n) \quad (4.7)$$

In this case, for any general triangulation \mathcal{T}_n divided, for example into n elements, a basis for $\mathbb{P}^0(\mathcal{T}_n)$ having n members can be constructed, one for each element in \mathcal{T}_n . A basis for $\mathbb{P}_{\text{cont}}^1(\mathcal{T}_n)$ constructed of hat functions would have $n + 1$ members, one for each ‘node point’ (or *vertex*) in \mathcal{T}_n . However, with the boundary condition $v_h|_{\Gamma_+} = 0$ we see that the same set excluding the member centered at Γ_+ is also a valid basis. The resultant set would have n elements. Thus, we can in general have a Petrov-Galerkin application, since we require $\dim(\mathbb{U}_h) = \dim(\mathbb{V}_h)$.

However, in two dimensions, the process is not so straightforward. If we wish to define $\mathbb{U}_h, \mathbb{V}_h$ on the same triangulation \mathcal{T}_n , for general inflow boundary (and therefore general curvilinear β), we would be restricted to those meshes and flows in which the number of vertices, excluding the vertices that lie on Γ_+ , is coincidentally equal to the number of elements in the mesh. It is not difficult to convince oneself that this is unlikely in general. Additionally, one might note that in the resultant system of linear equations, the number of equations is dictated by $\dim(\mathbb{V}_h)$, and the number of unknowns by $\dim(\mathbb{U}_h)$. For this reason, it is clear that any method we generate requires *at least* $\dim(\mathbb{U}_h) \leq \dim(\mathbb{V}_h)$.

For Petrov-Galerkin, there are methods to circumvent this problem. One way would be to choose a triangulation \mathcal{T}_n such that the resultant mesh is ‘flow-aligned’ for a given β . This allows the mesh to be partitioned (into say, m submeshes \mathcal{T}_n^m) such that on each partition, $\dim(\mathbb{U}_h^i) = \dim(\mathbb{V}_h^i)$ ($i = 1, \dots, m$), and as such the method can be performed on each submesh independently. However, this will naturally introduce discontinuities in the \mathbb{P}^1 test space across the submesh boundaries. Another way is to build the spaces on different meshes, such that the test space mesh is a conformal refinement of the trial

space mesh (i.e $\mathbb{U}_h = \mathbb{P}^0(\mathcal{T}_n)$, $\mathbb{V}_h = \mathbb{P}_{\text{cont}}^1(R(\mathcal{T}_n))$, where $R(\cdot)$ is an operator defining some refinement of the mesh). Building one as a refinement of the other is not technically necessary, but is extremely useful for practical calculation purposes. It is likely that for any mesh, we can construct some arbitrary refinement that would introduce exactly the correct number of degrees of freedom to make the system square, but this solution is inelegant, situationally dependent and impractical. A superior method would allow us to take any mesh, for any flow, apply some *structured* refinement to it, and still be able to solve the resultant system. Petrov-Galerkin by itself is inadequate for this purpose.

4.2.2 The Discrete-Dual Minimal-Residual Method

The Discrete-Dual Minimal-Residual method (DDMRes) can be thought of as a generalisation of Petrov-Galerkin to discrete formulations where $\dim(\mathbb{U}_h) \neq \dim(\mathbb{V}_h)$. The scheme introduces the discrete *residual*

$$\rho_h = f - Aw_h \in (\mathbb{V}_h)^* \quad (4.8)$$

which is, in some sense, a measure of the inability of an attempt, w_h to solve the problem. The idea is to minimise ρ_h with respect to the norm of $(\mathbb{V}_h)^*$, the dual of the finite dimensional subspace of \mathbb{V} . This norm is defined (for some $\psi_h \in (\mathbb{V}_h)^*$) as

$$\|\psi_h\|_{(\mathbb{V}_h)^*} = \sup_{v_h \in \mathbb{V}_h} \frac{\langle \psi_h, v_h \rangle_{\mathbb{V}^*, \mathbb{V}}}{\|v_h\|_{\mathbb{V}}}. \quad (4.9)$$

If we have the pairings as defined in (2.3), we may write

$$\|\rho_h\|_{(\mathbb{V}_h)^*} = \sup_{v_h \in \mathbb{V}_h} \frac{l(v_h) - b(w_h, v_h)}{\|v_h\|_{\mathbb{V}}}. \quad (4.10)$$

We now seek the solution as the argument that produces the minimum residual over all w_h , and our problem is defined:

Find $u_h \in \mathbb{U}_h$ s.t.

$$u_h = \arg \min_{w_h \in \mathbb{U}_h} \|\rho_h\|_{(\mathbb{V}_h)^*}. \quad (4.11)$$

It can be shown (see A.9 for a proof) that this problem is equivalent to the following:

Find $u_h \in \mathbb{U}_h$, $r_h \in \mathbb{V}_h$ s.t.

$$(r_h, v_h)_{\mathbb{V}} + b(u_h, v_h) = l(v_h) \quad \forall v_h \in \mathbb{V}_h \quad (4.12a)$$

$$b(w_h, r_h) = 0 \quad \forall w_h \in \mathbb{U}_h. \quad (4.12b)$$

The well-posedness of the problem in this form was shown in [1], Thm. B, from which we also recover the a priori error estimate

$$\|u - u_h\|_{L^2(\Omega)} \leq C \inf_{w_h \in \mathbb{U}_h} \|u - w_h\|_{L^2(\Omega)}. \quad (4.13)$$

Following from (4.9), where \mathbb{V} is the graph space $W^+(\boldsymbol{\beta}; \omega)$, we obtain the system of equations:

$$\begin{aligned} R_i \int_{\Omega} \nabla \cdot (\boldsymbol{\beta} \phi_i) \nabla \cdot (\boldsymbol{\beta} \phi_j) d\Omega + U_k \int_{\Omega} \nabla \cdot (\boldsymbol{\beta} \phi_i) \psi_k d\Omega \\ = \int_{\Omega} f \phi_i d\Omega + \int_{\Gamma_-} g \phi_i |\boldsymbol{\beta} \cdot \hat{\mathbf{n}}| d\Gamma_-, \\ \int_{\Omega} \nabla \cdot (\boldsymbol{\beta} \phi_j) \psi_k d\Omega = 0 \quad i, j = 1, \dots, \dim(\mathbb{V}_h), \quad k = 1, \dots, \dim(\mathbb{U}_h) \end{aligned}$$

This can be arranged into a single system $\mathbf{A}\mathbf{X} = \mathbf{F}$, where

$$\begin{aligned} \mathbf{A} &= \begin{pmatrix} \mathbf{B} & \mathbf{C} \\ \mathbf{C}^T & \mathbf{0} \end{pmatrix}, \quad \mathbf{X} = \begin{pmatrix} \mathbf{R} \\ \mathbf{U} \end{pmatrix}, \quad \mathbf{F} = \begin{pmatrix} \mathbf{L} \\ \mathbf{0} \end{pmatrix}, \\ B_{ij} &= \int_{\Omega} \nabla \cdot (\boldsymbol{\beta} \phi_i) \nabla \cdot (\boldsymbol{\beta} \phi_j) d\Omega, \\ C_{ki} &= \int_{\Omega} \nabla \cdot (\boldsymbol{\beta} \phi_i) \psi_k d\Omega, \\ L_i &= \int_{\Omega} f \phi_i d\Omega + \int_{\Gamma_-} g \phi_i |\boldsymbol{\beta} \cdot \hat{\mathbf{n}}| d\Gamma_-, \end{aligned}$$

and (\mathbf{R}, \mathbf{U}) are the column vectors representing the coefficients for (r_h, u_h) respectively. We note that in the case $\dim(\mathbb{U}_h) = \dim(\mathbb{V}_h)$, $\mathbf{R} = \mathbf{0}$ and we are performing a Petrov-Galerkin scheme. This is why DDMRes can be thought of as an extension to Petrov-Galerkin methods.

In our Implementation, we will be using the trial space $\mathbb{U}_h = \mathbb{P}^0(\mathcal{T}_n)$ - the set of piecewise constants defined on our triangulation \mathcal{T}_n . Our test space will be $\mathbb{V}_h = \mathbb{P}_{\text{cont}}^1(R(\mathcal{T}_n))$ - the set of continuous piecewise linear functions defined on a refinement of the triangulation. In this scenario (and in particular, due to the piecewise constant approximation of the solution u), we are expecting approximability of the form:

$$\|u - u_h\|_{L^2(\Omega)} \leq ch \|u'\|_{L^2(\Omega)} \quad (4.14)$$

as shown by [4], Corollary 1.122. As a result, we note an optimal convergence rate in this setting constitutes $\mathcal{O}(h)$ dependence, and we broadly expect to observe this in any calculations.

4.3 Implementation

4.3.1 The Peterson Mesh

The two-dimensional triangulation \mathcal{T}_n we will be using for our trial space is the so-called Peterson Mesh ([10]), which is best defined graphically:

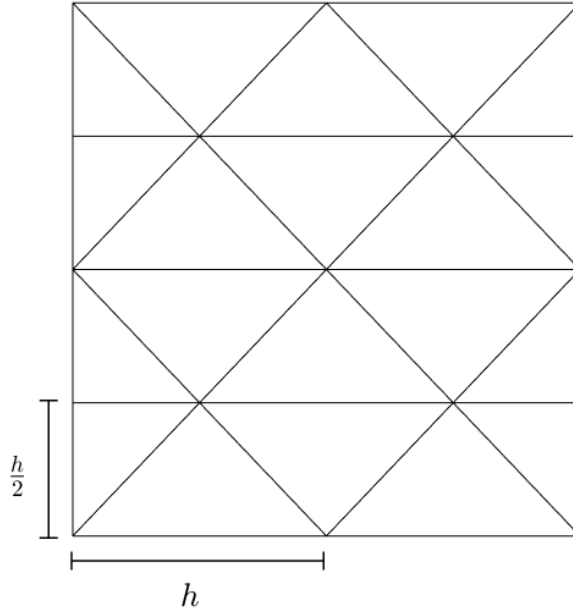


Figure 5: The Peterson Mesh on the unit square with degree $n = 2$, defined for $n = \frac{1}{h}$ (such as to be the number of partitions of the horizontal boundaries, analogously to the one-dimensional case.)

In two dimensions, elements are not defined along a straight line, but rather as a connection of three members of the set of vertices associated with the mesh. As a result, any two-dimensional mesh is necessarily defined by two matrices: \mathcal{V} (a *Vertex Array*) - a $2 \times N_{verts}$ -sized matrix, where each row i contains the (x, y) co-ordinates of the i^{th} vertex (ordered in some way) - and \mathcal{C} (a *Connectivity Array*) - a $3 \times N_{elems}$ -sized matrix, where each row j contains the 3 vertices that make up the j^{th} element, in terms of i (its corresponding row index in \mathcal{V}). The ordering of the vertices and elements in these arrays can be arbitrary, but when implementing a finite element scheme, it is beneficial to do so in some structured way that the implementation can then exploit. For our purposes, we will be ordering each vertex or element from left-to-right, and then from bottom to top (see Figure 6).

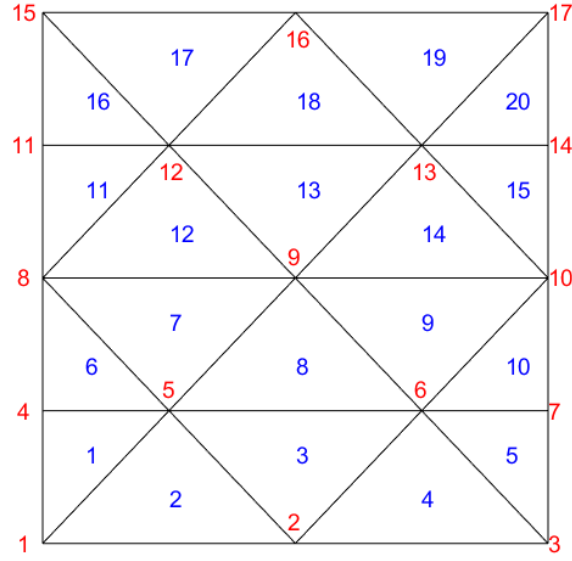


Figure 6: A numbering convention of the Peterson mesh - Red numbers denote vertices, blue numbers denote elements.

4.3.2 Mesh Refinement

To refine a Peterson mesh, I propose two techniques. The first is the so-called ‘red’ refinement scheme that can be applied to any mesh. It loops over the elements, and divides each one into four smaller triangles such that all they all have the same area, and the vertices of the center triangle is defined by the midpoints of each edge. Since this refinement works for any mesh, it can be applied an arbitrary number of times to the same mesh if one application does not merit sufficient resolution (see Figure 7).

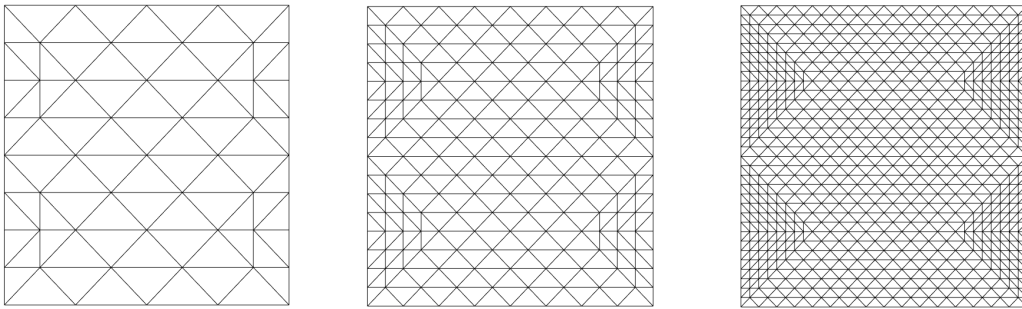


Figure 7: Procedural refinements of the $n = 2$ Peterson Mesh. The same numbering system for vertices/elements is applied to these.

The second refinement is specific to the Peterson Mesh, and is simply the addition of vertical lines every $\frac{h}{2}$ units to the mesh (see Figure 8). While it does not have implementation possibilities for general meshes, the specific nature of the refinement makes it useful for the cases where the flow is precisely constant in one of the four cardinal directions relative to the axes.

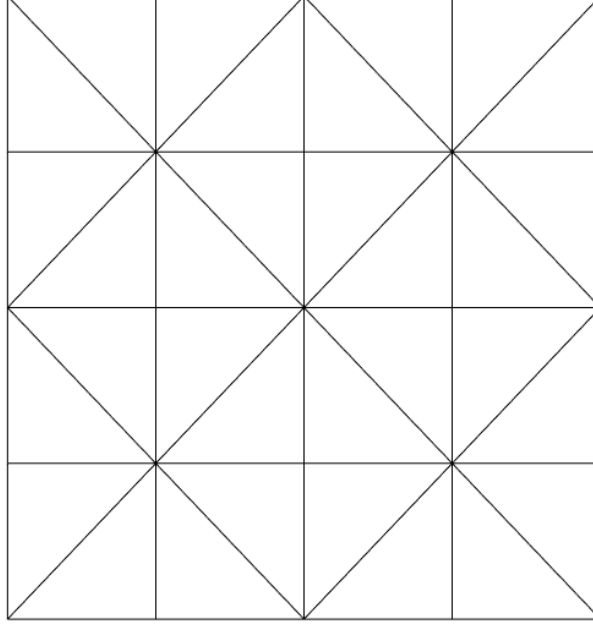


Figure 8: Vertical Refinement for $n = 2$ on the Peterson Mesh. Note this has the effect of partitioning the mesh into $4n^2$ squares of width $\frac{h}{2}$.

4.3.3 A Basis for the Trial and Test Spaces

The basis for our trial space, $\mathbb{U}_h = \mathbb{P}^0(\mathcal{T}_n)$, is a set of functions on each element τ_i , $i = 1, \dots, N_{elems}$ s.t.

$$\psi_i(x, y) = \begin{cases} 1 & (x, y) \in \tau_i \\ 0 & (x, y) \notin \tau_i \end{cases}$$

The test space, $\mathbb{V}_h = \mathbb{P}_{\text{cont}}^1(R(\mathcal{T}_n))$, has a more complicated basis, defined analogously to the ‘hat-functions’ (3.14) but in the two-dimensional sense. The most helpful way to define these functions (especially in terms of the computational application) for a given vertex \mathcal{V}_i is to consider a small, localised submesh of $R(\mathcal{T}_n)$ containing only those elements

that \mathcal{V}_i is associated with. For example, the basis function associated with \mathcal{V}_9 in Figure 6 would be nonzero only within $\{\tau_7, \tau_8, \tau_9, \tau_{12}, \tau_{13}, \tau_{14}\}$. These elements would be defined in terms of a *local* vertex and connectivity array. The example function is built such that it is exactly 1 at \mathcal{V}_9 , and decreases linearly along the lines connecting it to all adjacent vertices $\{\mathcal{V}_5, \mathcal{V}_6, \mathcal{V}_8, \mathcal{V}_{10}, \mathcal{V}_{12}, \mathcal{V}_{13}\}$, where it is 0. The function value inside the elements is defined by the planes that connect the linear functions on the edges, and 0 everywhere outside the submesh. The resultant shape is ‘pyramid-like’ (see Figure 9).

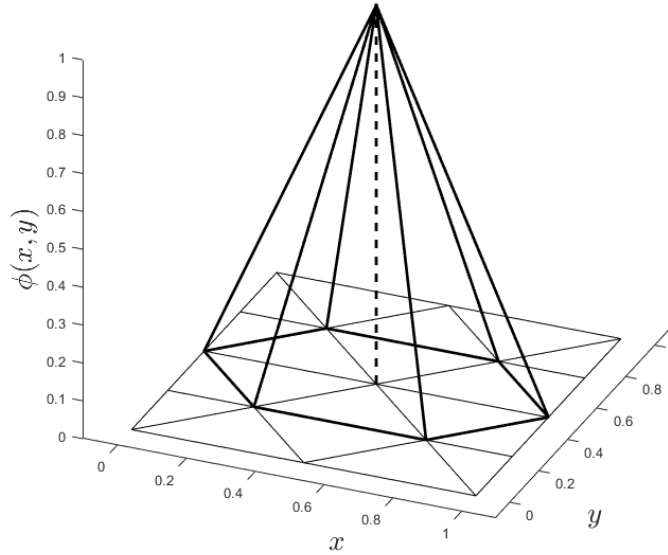


Figure 9: The Basis Function for \mathcal{V}_9 in the $n = 2$ Peterson Mesh.

We have in total N_{verts} functions that can be defined on the mesh, but since we have $\mathbb{P}_{\text{cont}}^1(R(\mathcal{T}_n)) \subset W_+(\boldsymbol{\beta}; \Omega)$, we must necessarily exclude those basis functions on vertices that lie on the outflow boundary, the number of which we shall call $N_{\text{outflow}}(R(\mathcal{T}_n))$. To make the system solvable, we need to satisfy $\dim(\mathbb{V}_h) \geq \dim(\mathbb{U}_h)$, and so require

$$N_{\text{verts}}(R(\mathcal{T}_n)) - N_{\text{outflow}}(R(\mathcal{T}_n)) \geq N_{\text{elems}}(\mathcal{T}_n) \quad (4.15)$$

It can be shown that for just a single application of the red refinement scheme, we satisfy (4.15) for all $n, \boldsymbol{\beta}$, whereas the ‘vertical line’ refinement achieves strict equality in (15) for

the case where β is precisely $(0, \alpha)^T$ or $(\alpha, 0)^T \forall \alpha \in \mathbb{R}$. Proofs of these propositions can be found in the appendix (A.10 and A.11).

4.3.4 Code Implementation and Class Structure

From a coding implementation standpoint, the system takes the form of several interacting classes, each with its own properties and methods. Some of these classes were built for more general use. For example, the main code is built to handle any general mesh, not just the Peterson meshes, and any refinement on that mesh, as long as the resultant system is solvable.

To handle basic Matrix manipulation, I developed a class to handle Matrices, housing the Gaussian Elimination method that is used to solve the resultant system. Similarly, a class to handle Vectors was used. These two classes are effectively the base code on which the two next classes are built.

One of these is a class to store meshes, in terms of their vertex and connectivity arrays. Associated with this mesh is a red refinement scheme, which can be applied an arbitrary number of times to any mesh. Also built was a ‘vertical lines’ refinement method, which can only be applied specifically to a Peterson mesh, and only once. Additionally on this class, I designed other methods to aid the construction of the system, including access methods for the connectivity/vertex arrays in terms of their index.

The other class stored the information associated with the $\mathbb{P}_{\text{cont}}^1$ basis functions as described in Figure 9. The information is stored in terms of a Matrix of co-ordinates, the first being the center vertex, and the remaining points denote the surrounding vertices in a structured way. Since these are being built on some Mesh object, we also have a local connectivity matrix associated with this class, to describe which combination of these vertices make up the elements in the underlying mesh. Among other methods developed to make the computations more clear (like calculating areas, evaluating particular integrals, returning the gradient in a given direction or the value of the basis function at a given point) is a unique method to resolve whether or not a certain point in two-dimensional space lies within a triangle, taken from [11].

The bulk of the main operation routine works as follows:

- Initialise Peterson mesh of degree n , and refine with one of the two refinement schemes.
- Initialise an array of basis function objects with size equal to the number of vertices in the test space.
- Loop over the objects, assigning their center and external vertices in terms of their local connectivity arrays.
- Initialise trial space with Peterson mesh of degree n .
- Initialise the matrix to house \mathbf{B} , and loop over the basis function array twice to set its values in terms of the relevant integrals.
- Initialise the matrix to house \mathbf{C} , and loop over the basis function array with the trial space connectivity array to set its values in terms of those integrals
- Combine these two Matrices to obtain \mathbf{A} .
- Remove the unnecessary indices of \mathbf{A} associated with outflow vertices
- Initialise \mathbf{F} , looping over the basis function array to assign values in terms of the f integrals.
- Loop over the basis functions associated with inflow vertices and perform the line integrals associated with g to add to \mathbf{F} .
- Remove the indices of the \mathbf{F} associated with outflow vertices.
- Initialise an empty solution vector \mathbf{U} and perform Gaussian Elimination on the system $\mathbf{AU} = \mathbf{F}$ to solve the system.

Focusing on the efficiency of the main routine is useful, but the computation time of the whole process is almost entirely dictated by the solution algorithm. The Gaussian Elimination algorithm runs at effectively $\mathcal{O}(n^5)$ in terms of the degree n of the Peterson mesh. This dependence can be mediated with the introduction of selective refinement methods (as we will see in section 4.4), but meaningful progress for the efficiency of this

routine would necessarily depend on the inception of a different solution procedure, perhaps one that exploits the properties of the system to reduce the number of calculations.

4.4 Computations and Analysis

Computations were performed exclusively with constant β on pairs of Peterson meshes of degree $n = 2^\alpha$ with their corresponding refinement in either the red or vertical line refinement schemes. Errors in the solution were measured in the L^2 -norm of $u - u_h$ and $\nabla \cdot (\beta r_h)$, both of which are expected to demonstrate $\sim \mathcal{O}(h)$ convergence. Since the solution u_h is given in terms of a piecewise constant basis, a MATLAB scheme was used to interpolate the data onto a uniform grid before display. This interpolation is performed after the norms are calculated, so are purely for aesthetic purposes and do not affect the data on display in this section.

4.4.1 Test Cases Demonstrating $\mathcal{O}(h)$ Convergence

Initial tests of the implementation were restricted to cases where either $f = 0$ or $g = 0$, and with $\beta = (0, 1)^T$. The first allows $f = 0$ by setting

$$u(x, y) = \sin(\pi x) \rightarrow g = \sin(\pi x)$$

Here, β and g do not interact, so the resultant system is functionally identical (as long as $\beta_1 = 0$). The plots for 4-16 are interpolated onto a 50×50 grid for the red refinement (see Figure 10) and the vertical line refinement (Figure 11). The data on the Errors is shown in Tables 1 and 2. For both we can see $\mathcal{O}(h)$, optimal for this discretisation. The efficiency bonus afforded to the vertical refinement scheme was the motivation for its design. Because the solver performing Gaussian Elimination is an $\mathcal{O}(m^3)$ process, where m is the number of equations, and doubling n produces $\mathcal{O}(n^2)$ equations, we expect our computation time to increase by a factor of $\sim 2^5$ upon doubling n . This is observed in both tables. The next test case (to allow $g = 0$) will be

$$u(x, y) = e^x(1 - \cos(y)) \rightarrow f(x, y) = e^x \sin(y)$$

Similarly, the data is displayed for both the red refinement (Figure 12 and Table 3) and the vertical refinement schemes (Figure 13 and Table 4). A similar ‘crinkling effect’ is observed here, and again we see $\mathcal{O}(h)$ convergence in both norms, with an increase in the efficiency for the vertical line refinement and less pronounced artifacts.

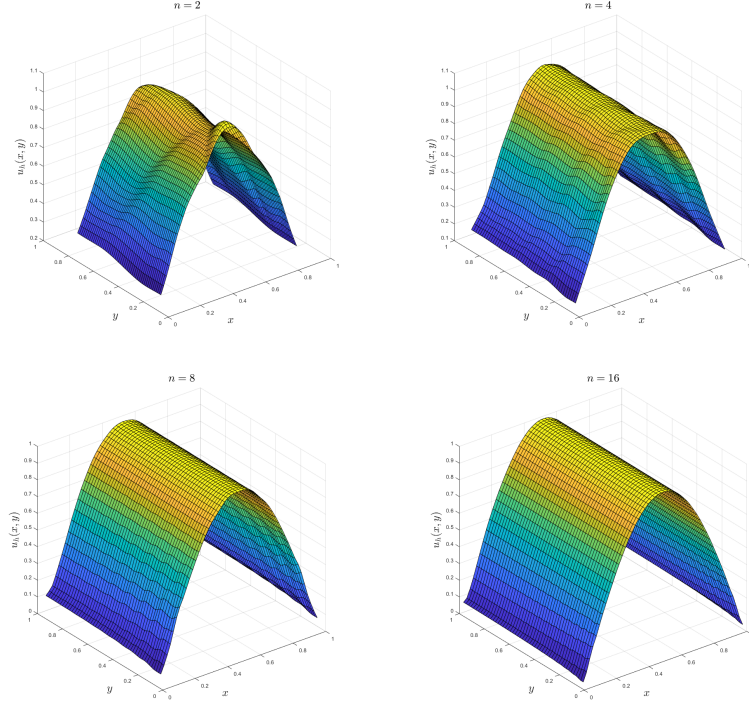


Figure 10: $u = \sin(\pi x)$ approximated on the Peterson mesh and red refinement scheme for $n = 2, 4, 8, 16$. Note the ‘paper crinkling effect’, less pronounced as n doubles. These artifacts are likely a combination of the lack of resolution in the trial space combined with the interpolation being performed by MATLAB.

n	$\ (u - u_h)\ _{L^2}$	Error Ratio	$\ \nabla \cdot (\beta r_h)\ _{L^2}$	Error Ratio	Time Taken (s)
1	0.26475	N/A	0.209345	N/A	0.0099975
2	0.186375	1.42052	0.145107	1.4427	0.0329901
4	0.107935	1.72674	0.0848735	1.70968	0.189938
8	0.0566188	1.90634	0.0454206	1.86861	3.00004
16	0.0289472	1.95593	0.0234832	1.93418	136.699

Table 1: Data on the errors and time taken to compute the approximations to $u = \sin(\pi x)$ with the red refinement scheme.

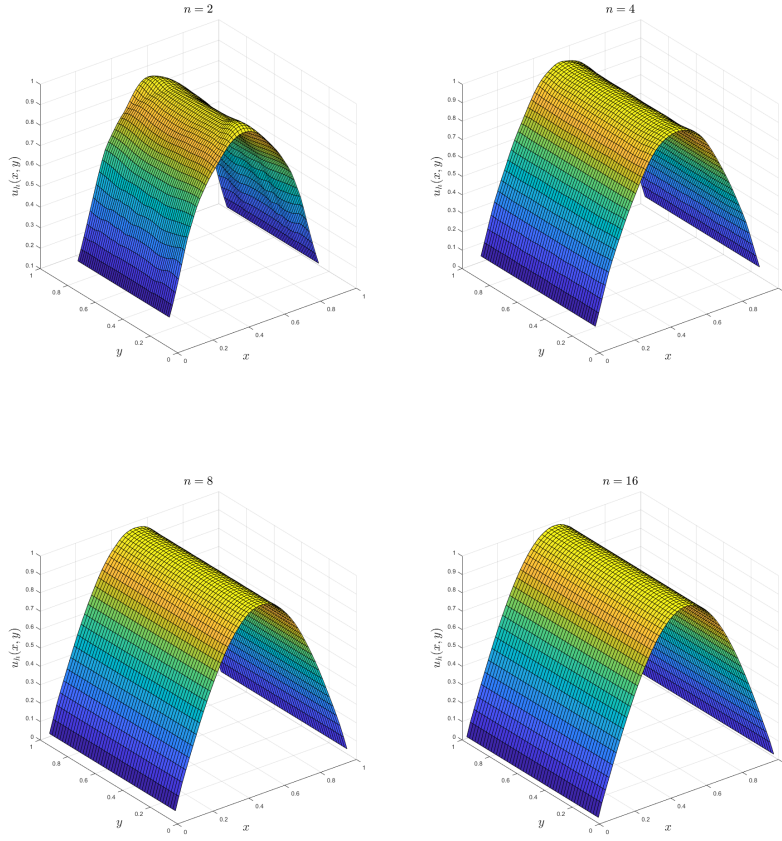


Figure 11: $u = \sin(\pi x)$ approximated on the Peterson mesh and vertical line refinement scheme for $n = 2, 4, 8, 16$. The ‘paper crinkling’ is significantly less pronounced here.

n	$\ (u - u_h)\ _{L^2}$	Error Ratio	$\ \nabla \cdot (\beta r_h)\ _{L^2}$	Error Ratio	Time Taken (s)
1	0.274796	N/A	0	N/A	0.0059896
2	0.18606	1.47692	0	nan	0.011996
4	0.103903	1.7907	0	nan	0.109472
8	0.054416	1.90943	0	nan	0.992939
16	0.0277874	1.9583	0	nan	29.0517

Table 2: Data on the errors and time taken to compute the approximations to $u = \sin(\pi x)$ with the vertical line refinement scheme. Note the decreased computation time and 0 in the second norm - both are the result of the reduction of this method to a Petrov-Galerkin scheme (A.1).

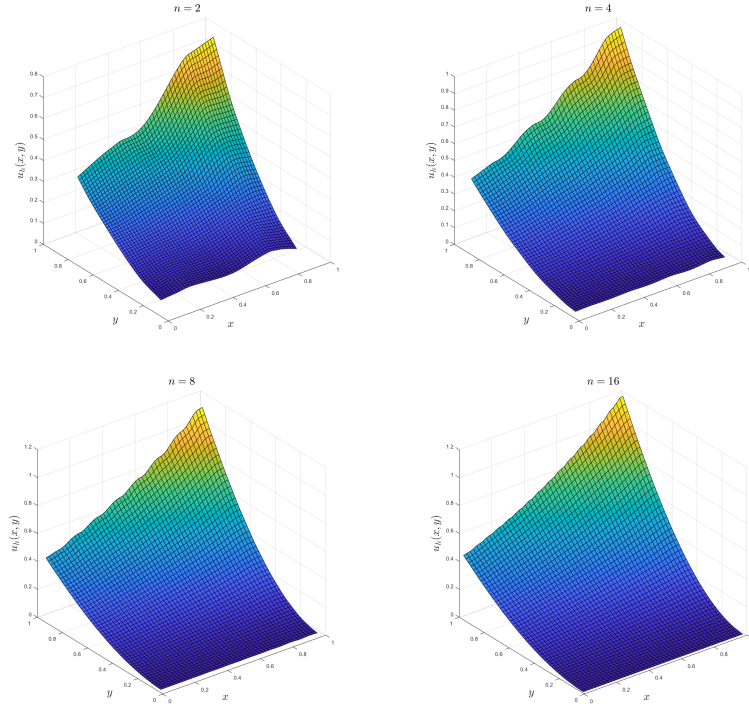


Figure 12: $u = e^x(1 - \cos(y))$ approximated on the Peterson mesh and red refinement scheme for $n = 2, 4, 8, 16$. The crinkling effect is less obvious but still apparent in the solution.

n	$\ (u - u_h)\ _{L^2}$	Error Ratio	$\ \nabla \cdot (\beta r_h)\ _{L^2}$	Error Ratio	Time Taken (s)
1	0.123497	N/A	0.106748	N/A	0.0336696
2	0.0677552	1.82269	0.052462	2.03476	0.0259918
4	0.034094	1.98731	0.0272727	1.92361	0.178954
8	0.0170218	2.00295	0.0140261	1.94443	2.94505
16	0.00849502	2.00374	0.00712532	1.96848	93.9316

Table 3: Data on the errors and time taken to compute the approximations to $u = e^x(1 - \cos(y))$ with the red refinement scheme.

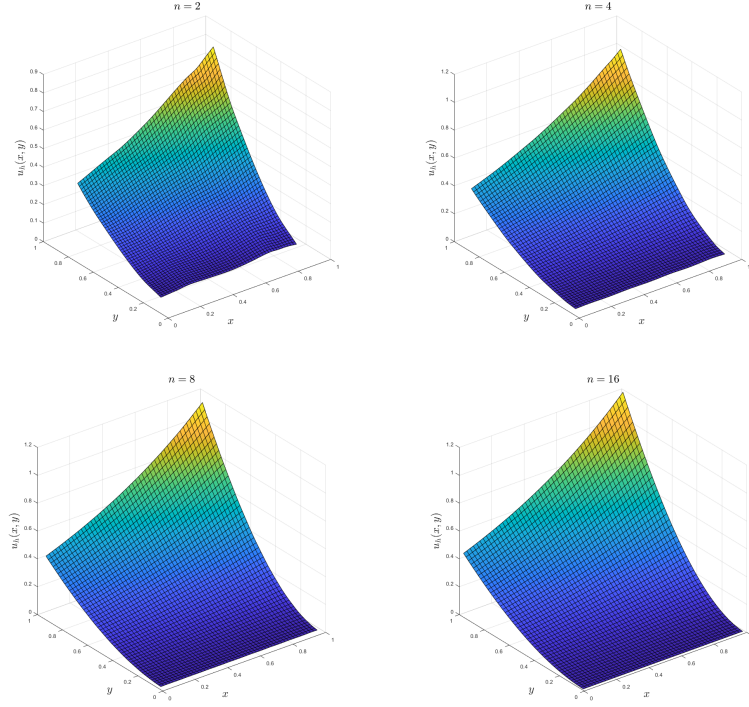


Figure 13: $u = e^x(1 - \cos(y))$ approximated on the Peterson mesh and vertical line refinement scheme for $n = 2, 4, 8, 16$. Again, the crinkling effect is less pronounced, and the solution becomes smooth much more quickly.

n	$\ (u - u_h)\ _{L^2}$	Error Ratio	$\ \nabla \cdot (\beta r_h)\ _{L^2}$	Error Ratio	Time Taken (s)
1	0.119126	N/A	0	N/A	0.0089972
2	0.0640472	1.85997	0	nan	0.0129952
4	0.0328671	1.94867	0	nan	0.0679782
8	0.0166132	1.97838	0	nan	0.955705
16	0.00834778	1.99013	0	nan	29.0215

Table 4: Data on the errors and time taken to compute the approximations to $u = e^x(1 - \cos(y))$ with the vertical line refinement scheme. Again, the computation time is superior to the red refinement. We also have reduced this problem to a Petrov-Galerkin scheme, as before.

4.4.2 Horizontal Flow - Singularity in the Vertical Refinement Scheme

If we take $u = e^x(1 - \cos(y))$ as before, but with $\beta = (\pm 1, 0)^T$, we get

$$f = \pm e^x(1 - \cos(y)), \quad g = (1 - \cos(y)) \text{ or } g = e(1 - \cos(y))$$

We see in the data for both directions, in the red refinement scheme, that we are still converging optimally in both norms (Table 5). However, a surprising result is that the vertical refinement scheme results in a singular matrix for horizontal flow. Since it was designed specifically to handle purely vertical or horizontal flow, the reason for this is unclear. On inspection of any example system, it is easy to see that two of the columns are actually identical, so the singularity is readily apparent. This may have something to do with an unfortunate combination of the symmetries of the mesh with the column/row exclusions in the system as a result of the boundary conditions, but further investigation would be required to confirm this.

n	$\ (u - u_h)\ _{L^2}$	Error Ratio	$\ \nabla \cdot (\beta r_h)\ _{L^2}$	Error Ratio	Time Taken (s)
1	0.130488	N/A	0.085587	N/A	0.0101011
2	0.0717243	1.81929	0.0375493	2.27932	0.0249919
4	0.0373774	1.91892	0.0176178	2.13133	0.198936
8	0.0190475	1.96232	0.00855332	2.05976	3.24599
16	0.00961001	1.98205	0.00421709	2.02825	104.762

Table 5: Data on the errors and time taken to compute the approximations to $u = e^x(1 - \cos(y))$ in the flow directly right/left case. The data for the errors in the norms is actually identical.

4.4.3 General Flow Direction

Our final system examination will be one employing an arbitrary flow direction and 0 in the source term. By specifying an arbitrary g over the inflow boundary, we expect to see the function translated through the unit square in the direction of β , and as such, the solution $u(x, y)$ will be constant on the streamlines. If we define $(x_0, 0)^T$ and $(0, y_0)^T$ as the two straight lines defining the inflow boundary, we can write:

$$u(x, y) = \begin{cases} g(x_0, x - y \frac{\beta_1}{\beta_2}(x - x_0)) & \text{if streamline begins on the } x_0 \text{ boundary} \\ g(y - x \frac{\beta_2}{\beta_1}(y - y_0), y_0) & \text{if streamline begins on the } y_0 \text{ boundary} \end{cases}$$

If we choose (e.g)

$$g = e^x \cos(\pi y), \quad \beta = (1, 1)^T$$

then we expect

$$u(x, y) = \begin{cases} \cos(\pi(x - y)) & \text{if streamline begins on the } x_0 \text{ boundary} \\ e^{y-x} & \text{if streamline begins on the } y_0 \text{ boundary} \end{cases}$$

n	$\ (u - u_h)\ _{L^2}$	Error Ratio	$\ \nabla \cdot (\beta r_h)\ _{L^2}$	Error Ratio	Time Taken (s)
1	0.478423	N/A	0.187811	N/A	0.0099968
2	0.246833	1.93824	0.0602879	3.11523	0.0369881
4	0.130464	1.89197	0.0247972	2.43123	0.211932
8	0.0669735	1.94799	0.00851919	2.91075	3.17598
16	0.0338256	1.97996	0.0029388	2.89887	106.032

Table 6: Data on the errors and time taken to compute the approximations to $g = e^x \cos(\pi y)$ translated through the unit square for $\beta = (1, 1)^T$.

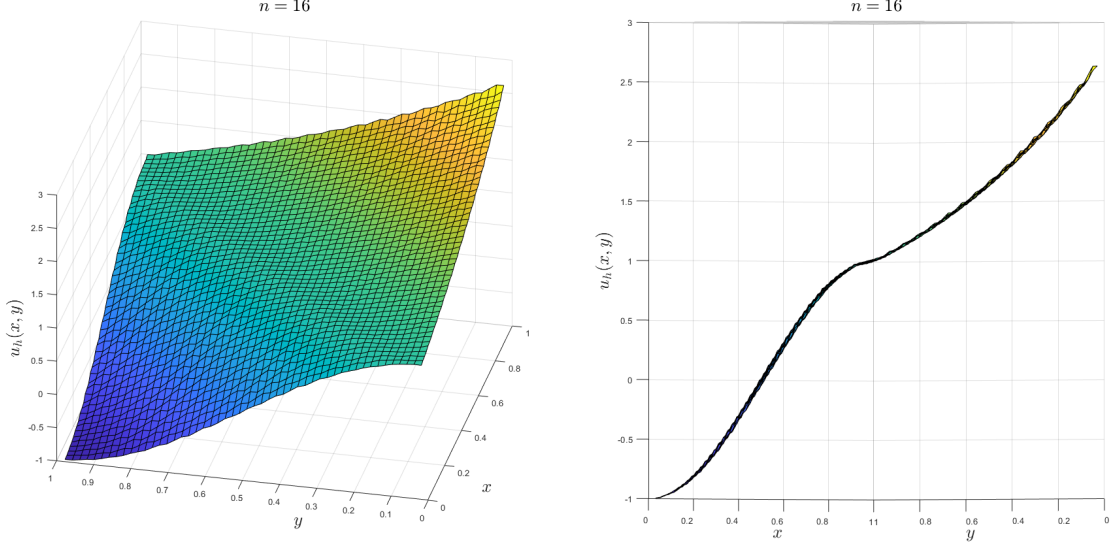


Figure 14: Two angles for the plot of the approximation to $g = e^x \cos(\pi y)$ translated through the unit square for $\beta = (1, 1)^T$, $n = 16$.

The plot of the solution u_h is shown for $n = 16$ in Figure 14, and the convergence data in Table 6. We see from the second image in Figure 14 how the solution remains constant on the streamlines, and is effectively one-dimensional given the correct orientation. We also again note optimal $\mathcal{O}(h)$ convergence in the L^2 -norm for $(u - u_h)$ but, strangely, $\sim \mathcal{O}(h^{1.585})$ convergence in the other. The reason for this is unclear.

Additionally, on computation of the solutions in the general flow direction case, I noticed an ambiguity in the treatment of the corner vertices of the mesh (or more precisely, ambiguity in the direction of its unit-outward normal, which is the source of this issue). For any corner vertex, one could ‘reasonably’ allow the unit-normal to be any value in the range of angles between the perpendiculars drawn to the two edges that define it. For any ‘non-cardinal’ flow, the two corners present ‘along’ the flow direction are unambiguously inflow/outflow vertices, in that any reasonable ‘choice’ of unit-outward normal vector would give the same answer in terms of (4.2). However, the other two can be a member of the inflow set, outflow set or the characteristic set depending on the precise definition of $\hat{\mathbf{n}}$. In this case, $\mathcal{O}(h)$ convergence is only observed in the results if one assumes that these corner vertices are in all three sets simultaneously! In the context of this implementation, it means that the relevant line integrals are calculated as if it were an inflow vertex, but

the index is excluded from the final system as if it were an outflow vertex. This is a surprising requirement, and is likely a consequence of the lack of ‘well-separated’ inflow and outflow in the case of any square mesh.

5 Conclusions

After a demonstration of the unsatisfactory approximation standard for solutions in first-order PDEs at the one-dimensional level, several more stable systems have been investigated. For one dimension we see that the least-squares method performs optimally in approximations of functions in H^2 , and still converges, albeit not optimally, to others in lower order Sobolev spaces. The system, obtained by setting up the weak form of the problem analogously to a second order PDE of similar form, demonstrates a way to consider a ‘classically’ non-coercive problem such as one-dimensional advection. For two-dimensional advection, testing of simple functions over the unit square with the Peterson mesh saw $\mathcal{O}(h)$ convergence in the L^2 -norm of the error, although both refinement methods demonstrated unusual ‘crinkling’ phenomena in the solution for lower resolutions. The vertical line refinement demonstrated an efficiency (and moderate accuracy) boost for a given degree of triangulation n in the flow-up case, but unexpectedly generates a singular system in the horizontal flow case, which was partial motivation for its design. The red refinement scheme also performs optimally in the constant β case for general angle, and converges to a similar order in the L^2 -norm for up to $n = 16$.

In this setting, the red refinement scheme performs well on the Peterson mesh in the constant flow case, and although it is inefficient (requiring more integrals to be calculated than are necessary), allowing $\dim \mathbb{U}_h \leq \dim \mathbb{V}_h$ is the useful feature of DDMRes in general. Boosts to performance efficiency can be found in specialised refinements on a case-by-case basis, but it is advantageous to have a scheme that will be able to find a procedural refinement of any mesh with a sufficient number of vertices.

The results provide an insight into the behaviour of the DDMRes method within the context of this simplified setting, and verify its capacity to converge on a given solution optimally, as a marked improvement over previous attempts to solve advection problems on the Peterson mesh (as with discontinuous Galerkin formulations).

The nature of this dissertation provides several directions for further study into the topic,

with an emphasis on investigating more general meshes (perhaps on convex polygonal domains), curvilinear flow, nonzero reaction coefficients, and investigating the approximation of discontinuities in the solution. In terms of solving the system, the key problem was the use of a general-purpose Gaussian Elimination scheme for a matrix system that was usually quite sparse. A significant performance boost to the method as performed here would be to investigate exploitable properties of the resultant systems such that a dedicated solution algorithm could be developed. If this was, say $\mathcal{O}(n^2)$, solutions of significantly higher resolution would be reachable with a reasonable computation time. The design of algorithms for the project was done in an intentionally general and modular way, such as to allow these extensions to become natural and allow the project to be as open-ended as possible.

A Appendix

A.1 Coercivity and Positive Definiteness

A Positive Definite matrix $A \in \mathbb{R}^d$ can be defined as one for which the scalar value

$$\mathbf{x}^T \mathbf{A} \mathbf{x} > 0 \quad \forall \mathbf{x} \neq \mathbf{0} \quad (\text{A.1})$$

and we may write this in terms of the summations performed as

$$\sum_i^d x_i \left(\sum_j^d A_{ij} x_j \right). \quad (\text{A.2})$$

On discretisation, our bilinear form will be written as $b(w_h, v_h)$ where $w_h, v_h \in \mathbb{U}_h \subset \mathbb{U}$. Therefore, we have $b(w_h, w_h) = c_0 \|w_h\|_{\mathbb{U}}^2 \geq 0$ from coercivity. Furthermore, we can write w_h as a linear combination of a set of basis functions for \mathbb{U}_h , and our matrix A_{ij} is equivalent to $b(\phi_i, \phi_j)$, where the ϕ are the basis functions. Thus, we find

$$\begin{aligned} b(w_h, w_h) &= b\left(\sum_i^d x_i \phi_i, \sum_j^d x_j \phi_j\right) \\ &= \sum_i^d x_i \left(\sum_j^d x_j b(\phi_i, \phi_j) \right) \\ &= \mathbf{x}^T \mathbf{A} \mathbf{x} \geq 0 \quad \forall \mathbf{x} \in \mathbb{R}^d \end{aligned} \quad (\text{A.3})$$

We now see that from coercivity, equality in (eqn A.3) is achieved iff the $\|w_h\|_{\mathbb{U}} = 0$, which occurs iff $w_h = 0 \rightarrow \mathbf{x} = 0$, and so finally we have

$$\mathbf{x}^T \mathbf{A} \mathbf{x} > 0 \quad \forall \mathbf{x} \neq \mathbf{0}. \quad \blacksquare$$

A.2 Generating Céa's Lemma from Well-Posedness Assumptions

The well-known Galerkin Orthogonality can be constructed from (2.3) and (2.5) like this:

Since $v_h \in \mathbb{V}$, we may write

$$\begin{aligned} b(u, v_h) &= l(v_h) = b(u_h, v_h) \\ \rightarrow b(u, v_h) - b(u_h, v_h) &= 0 \\ \rightarrow b(u - u_h, v_h) &= 0 \quad \forall v_h \in \mathbb{V}_h. \end{aligned} \tag{A.4}$$

Now we generate the inequality by observing that

$$\begin{aligned} \|u - u_h\|_{\mathbb{U}} &= \|u - w_h - (u_h - w_h)\|_{\mathbb{U}} \\ &\leq \|u - w_h\|_{\mathbb{U}} + \|u_h - w_h\|_{\mathbb{U}} \quad \forall w_h \in \mathbb{U}_h. \end{aligned} \tag{A.5}$$

The discrete inf-sup condition tells us that

$$\begin{aligned} \sup_{v_h \in \mathbb{V}_h} \frac{b(w_h, v_h)}{\|v_h\|_{\mathbb{V}}} &\geq \hat{\gamma} \|w_h\|_{\mathbb{U}} \quad \forall w_h \in \mathbb{U}_h \\ \rightarrow \sup_{v_h \in \mathbb{V}_h} \frac{b(u_h - w_h, v_h)}{\|v_h\|_{\mathbb{V}}} &\geq \hat{\gamma} \|u_h - w_h\|_{\mathbb{U}} \quad \forall w_h \in \mathbb{U}_h. \end{aligned} \tag{A.6}$$

Using (A.4), we can write $b(u_h - w_h, v_h)$ as $b(u - w_h, v_h)$, and so we have

$$\begin{aligned} \|u - u_h\|_{\mathbb{U}} &\leq \|u - w_h\|_{\mathbb{U}} + \frac{1}{\hat{\gamma}} \sup_{v_h \in \mathbb{V}_h} \frac{b(u - w_h, v_h)}{\|v_h\|_{\mathbb{V}}} \\ &\leq \|u - w_h\|_{\mathbb{U}} + \frac{1}{\hat{\gamma}} \sup_{v_h \in \mathbb{V}_h} \frac{c_1 \|u - w_h\|_{\mathbb{U}} \|v_h\|_{\mathbb{V}}}{\|v_h\|_{\mathbb{V}}} \\ &\leq \left(1 + \frac{c_1}{\hat{\gamma}}\right) \|u - w_h\|_{\mathbb{U}} \quad \forall w_h \in \mathbb{U}_h \\ \rightarrow \|u - u_h\|_{\mathbb{U}} &\leq \inf_{w_h \in \mathbb{U}_h} \left(1 + \frac{c_1}{\hat{\gamma}}\right) \|u - w_h\|_{\mathbb{U}}. \end{aligned} \quad \blacksquare$$

A.3 Constructing the Standard Galerkin System in One Dimension

Since $\text{supp}(\phi_i) = \text{supp}(\phi'_i) = [x_{i-1}, x_{i+1}]$ of width $2h$ in the integrand, we note that $A_{ij} = 0$ if $|i - j| > 1$. Thus, the system is tridiagonal and we have 3 cases to investigate. If $i = j$:

$$\begin{aligned}
 A_{ij} &= \int_{\Omega} \phi_i \phi'_i dx \\
 &= \int_{x_i}^{x^{i+1}} \phi_i \phi'_i dx + \int_{x_{i-1}}^{x^i} \phi_i \phi'_j dx \\
 &= \frac{1}{h} \int_{x_i}^{x^{i+1}} \phi_i dx - \frac{1}{h} \int_{x_{i-1}}^{x^i} \phi_i dx \\
 &= 0 \quad (\phi_i \text{ symmetrix about } x = x_i).
 \end{aligned} \tag{A.7}$$

If $i = j - 1$

$$\begin{aligned}
 A_{ij} &= \int_{\Omega} \phi_i \phi'_j dx \\
 &= \int_{x_i}^{x^{i+1}} \phi_i \phi'_i dx \\
 &= \frac{1}{h} \int_{x_i}^{x^{i+1}} \phi_i dx \\
 &= \frac{1}{h} \left(\frac{h}{2} \right) = \frac{1}{2}.
 \end{aligned} \tag{A.8}$$

If $i = j + 1$

$$\begin{aligned}
 A_{ij} &= \int_{\Omega} \phi_i \phi'_j dx \\
 &= \int_{x_{i+1}}^{x^i} \phi_i \phi'_i dx \\
 &= -\frac{1}{h} \int_{x_i}^{x^{i+1}} \phi_i dx \\
 &= -\frac{1}{h} \left(\frac{h}{2} \right) = -\frac{1}{2}.
 \end{aligned} \tag{A.9}$$

And so we obtain \mathbf{A} as seen in (3.8). ■

A.4 A Proof of the Singularity of (3.8) in the Odd Dimensional Case

If we consider any $n \times n$ matrix A with only nonzero entries in the upper and lower diagonal such that (for, e.g, $n = 5$)

$$\begin{pmatrix} 0 & \alpha & 0 & 0 & 0 \\ -\alpha & 0 & \alpha & 0 & 0 \\ 0 & -\alpha & 0 & \alpha & 0 \\ 0 & 0 & -\alpha & 0 & \alpha \\ 0 & 0 & 0 & -\alpha & 0 \end{pmatrix}. \quad (\text{A.10})$$

We recall that if \mathbf{A} is invertible, then its columns are linearly independent of one another, so writing one column as a linear combination of the other is equivalent to showing a singular matrix. We can write the columns (denoted \mathbf{C}_i) in terms of their basis vectors $\hat{\mathbf{e}}_j$ and see that in general we have

$$\mathbf{C}_i = \begin{cases} \alpha \hat{\mathbf{e}}_{i-1} - \alpha \hat{\mathbf{e}}_{i+1} & i \neq 1, n \\ -\alpha \hat{\mathbf{e}}_{i+1} & i = 1 \\ \alpha \hat{\mathbf{e}}_{i-1} & i = n. \end{cases} \quad (\text{A.11})$$

If n is odd, we can write it as $2m + 1$, where m is even. If we now consider the negative of the sum of the odd-numbered columns up to but not including n , we see:

$$\begin{aligned} -\sum_{i=1}^m \mathbf{C}_{2i-1} &= -(\mathbf{C}_1 + \mathbf{C}_3 + \dots + \mathbf{C}_{2m-3} + \mathbf{C}_{2m-1}) \\ &= -\alpha(-\hat{\mathbf{e}}_2 + \hat{\mathbf{e}}_2 - \hat{\mathbf{e}}_4 + \hat{\mathbf{e}}_4 + \dots + \hat{\mathbf{e}}_{2m-4} - \hat{\mathbf{e}}_{2m-2} + \hat{\mathbf{e}}_{2m-2} - \hat{\mathbf{e}}_{2m}) \\ &= \alpha \hat{\mathbf{e}}_{2m} \\ &= \mathbf{C}_{2m+1}. \end{aligned} \quad \blacksquare$$

A.5 The Equivalence of Problems (3.10) and (3.13)

We begin with the equation (3.10), and assume u denotes the unique solution. Then,

$\forall w \in \mathbb{H}_0^1$:

$$\begin{aligned}
J(w) - J(u) &= \frac{1}{2}\tilde{b}(w, w) - \frac{1}{2}\tilde{b}(u, u) + \tilde{l}(u - w) \\
&= \frac{1}{2}\tilde{b}(w, w) - \frac{1}{2}\tilde{b}(u, u) + \tilde{b}(u, u - w) \\
&= \frac{1}{2}\tilde{b}(w, w) - \frac{1}{2}\tilde{b}(u, u) + \tilde{b}(u, u) - \tilde{b}(u, w) \\
&= \frac{1}{2} \left(\tilde{b}(w, w) + \frac{1}{2}\tilde{b}(u, u) + 2\tilde{b}(u, w) \right) \\
&= \frac{1}{2} \left(\tilde{b}(w, w) + \frac{1}{2}\tilde{b}(u, u) + \tilde{b}(u, w) - \tilde{b}(u, w) \right) \\
&= \frac{1}{2} \left(\tilde{b}(w - u, w) + \frac{1}{2}\tilde{b}(u, u - w) \right) \\
&= \frac{1}{2} \left(\tilde{b}(w - u, w - u) \right) \\
&\geq \frac{c_0}{2} \|w - u\|_H^1(\Omega) \\
&\geq 0.
\end{aligned} \tag{A.12}$$

If this is the case, then we have $J(w) \geq J(u) \ \forall w$, or equivalently

$$u = \arg \min_{w \in \mathbb{H}_0^1} J(w).$$

■

A.6 Lax Milgram Criteria Verification for (3.10)

Since it is trivial to show bilinearity for $\tilde{b}(\cdot, \cdot)$ and linearity for $\tilde{l}(\cdot)$, we will simply demonstrate the continuity of both operators for completeness.

For continuity of $\tilde{b}(\cdot, \cdot)$, we have:

$$\begin{aligned}
 |b(w, v)| &= \left| \int_{\Omega} w'(x) v'(x) \right| \\
 &\leq \|w'\|_{L^2(\Omega)} \|v'\|_{L^2(\Omega)} \\
 &\leq (\|w'\|_{L^2(\Omega)}^2 \|w\|_{L^2(\Omega)}^2)^{\frac{1}{2}} + (\|v'\|_{L^2(\Omega)}^2 + \|v'\|_{L^2(\Omega)}^2)^{\frac{1}{2}} \\
 &\leq \|w\|_{H^1(\Omega)} \|v\|_{H^1(\Omega)}
 \end{aligned} \tag{A.13}$$

and so the constant $c_1 = 1$.

For continuity of $\tilde{l}(\cdot)$, we have:

$$\begin{aligned}
 |l(v)| &= \left| \int_{\Omega} f(x) v'(x) \right| \\
 &\leq \|f\|_{L^2(\Omega)} \|v'\|_{L^2(\Omega)} \\
 &\leq \|f\|_{L^2(\Omega)} (\|v'\|_{L^2(\Omega)}^2 + \|v'\|_{L^2(\Omega)}^2)^{\frac{1}{2}} \\
 &= \|f\|_{L^2(\Omega)} \|v\|_{H^1(\Omega)}.
 \end{aligned} \tag{A.14}$$

and so the constant $c_2 = \|f\|_{L^2(\Omega)}$. ■

A.7 A Proof of the Discrete Inf-Sup Condition for (3.17)

As (3.10) and (3.11) are equivalent, the discretisation (on the same mesh \mathcal{T}_n) is an equivalent problem, and so we will focus on a proof of the discrete inf-sup condition for the problem:

Find $u_h \in \mathbb{P}_{\text{cont}}^1(\mathcal{T}_n)$ s.t.

$$\int_{\mathcal{T}_n} u'_h v_h dx = \int_{\mathcal{T}_n} f v_h dx \quad \forall v_h \in \mathbb{P}^0(\mathcal{T}_n) \quad (\text{A.15})$$

Thus, the condition we wish to prove is

$$\inf_{w_h \in \mathbb{P}_{\text{cont}}^1(\mathcal{T}_n)} \sup_{v_h \in \mathbb{P}^0(\mathcal{T}_n)} \frac{\int_{\mathcal{T}_n} w'_h v_h}{\|w'_h\|_{L^2} \|v_h\|_{L^2}} \geq \hat{\gamma} \quad (\text{A.16})$$

or equivalently:

$$\sup_{v_h \in \mathbb{P}^0(\mathcal{T}_n)} \frac{\int_{\mathcal{T}_n} w'_h v_h}{\|v_h\|_{L^2}} \geq \hat{\gamma} \|w'_h\|_{L^2} \quad \forall w_h \in \mathbb{P}_{\text{cont}}^1(\mathcal{T}_n). \quad (\text{A.17})$$

We note that we may choose $v_h = v_h(w_h) = w'_h$ for each case (since the derivative associated with the space $\mathbb{P}_{\text{cont}}^1(\mathcal{T}_n)$ is precisely $\mathbb{P}^0(\mathcal{T}_n)$). In this case, we find

$$\begin{aligned} \sup_{v_h \in \mathbb{P}^0(\mathcal{T}_n)} \frac{\int_{\mathcal{T}_n} w'_h v_h}{\|v_h\|_{L^2}} &\geq \frac{\int_{\mathcal{T}_n} w_h'^2}{\|w'_h\|_{L^2}} \\ &= \frac{\|w'_h\|_{L^2}^2}{\|w'_h\|_{L^2}} \\ &= \|w'_h\|_{L^2}. \end{aligned} \quad \blacksquare$$

A.8 Constructing the Least-Squares System in One Dimension

Similarly to (A.3), $\text{supp}(\phi'_i) = [x_{i-1}, x_{i+1}]$ of width $2h$ in the integrand, and so $A_{ij} = 0$ if $|i - j| > 1$. We again calculate the three cases independently.

If $i = j$:

$$\begin{aligned}
 A_{ij} &= \int_{\Omega} \phi'_i \phi'_i dx \\
 &= \int_{x_i}^{x^{i+1}} \phi'_i \phi'_i dx + \int_{x_{i-1}}^{x^i} \phi'_i \phi'_i dx \\
 &= \frac{1}{h^2} \int_{x_i}^{x^{i+1}} dx + \frac{1}{h^2} \int_{x_{i-1}}^{x^i} dx \\
 &= \frac{2}{h} \quad (\phi_i \text{ symmetrix about } x = x_i).
 \end{aligned} \tag{A.18}$$

If $i = j - 1$

$$\begin{aligned}
 A_{ij} &= \int_{\Omega} \phi'_i \phi'_j dx \\
 &= \int_{x_i}^{x^{i+1}} \phi'_i \phi'_i dx \\
 &= -\frac{1}{h^2} \int_{x_i}^{x^{i+1}} dx \\
 &= -\frac{1}{h}
 \end{aligned} \tag{A.19}$$

Similarly, if $i = j + 1$

$$\begin{aligned}
 A_{ij} &= \int_{\Omega} \phi'_i \phi'_j dx \\
 &= \int_{x_{i-1}}^{x^i} \phi'_i \phi'_i dx \\
 &= -\frac{1}{h^2} \int_{x_{i-1}}^{x^i} dx \\
 &= -\frac{1}{h}
 \end{aligned} \tag{A.20}$$

And so we obtain \mathbf{A} as seen in (3.17). ■

A.9 The Equivalence of Problems (4.11) and (4.12)

We begin with the introduction of a Riesz operator R_h , defined thusly:

$$R_h : \mathbb{V} \rightarrow (\mathbb{V})^* \text{ s.t.} \quad \langle R_h(v_h), \psi_h \rangle_{\mathbb{V}^*, \mathbb{V}} = (v_h, \psi_h)_{\mathbb{V}} \quad \forall \psi_h, v_h \in \mathbb{V}_h \quad (\text{A.21})$$

If we look now at the discrete dual norm of the residual ρ_h (as defined (eq number)), we may now write it as

$$\|\rho_h\|_{(\mathbb{V}_h)^*} = \sup_{\psi_h \in \mathbb{V}_h} \frac{\langle \rho_h, \psi_h \rangle}{\|\psi\|_{\mathbb{V}}} = \sup_{\psi_h \in \mathbb{V}_h} \frac{(R_h^{-1}(\rho_h), \psi)_{\mathbb{V}}}{\|\psi\|_{\mathbb{V}}} \quad (\text{A.22})$$

the supremum here is achieved when ψ_h lies precisely in the $R_h^{-1}(\rho_h)$ (i.e it can be written as $\alpha R_h^{-1}(\rho_h)$, where α is some scalar constant). Therefore we may write

$$\|\rho_h\|_{(\mathbb{V}_h)^*} = \frac{\alpha(R_h^{-1}(\rho_h), R_h^{-1}(\rho_h))_{\mathbb{V}}}{\alpha \|R_h^{-1}(\rho_h)\|_{\mathbb{V}}} = \frac{(R_h^{-1}(\rho_h), R_h^{-1}(\rho_h))_{\mathbb{V}}}{(R_h^{-1}(\rho_h), R_h^{-1}(\rho_h))_{\mathbb{V}}^{\frac{1}{2}}} = \|R_h^{-1}(\rho_h)\|_{\mathbb{V}}. \quad (\text{A.23})$$

For simplicity, we write $y = R_h^{-1}(f)$, $Bw_h = R_h^{-1}(Aw_h)$, $\hat{\rho}_h = y - Bw_h$. We note now that the minimisation problem (4.14) is equivalent to

$$\text{Find } u_h \in \mathbb{U}_h \text{ s.t.} \quad u_h = \arg \min_{w_h \in \mathbb{U}_h} \frac{1}{2} \|y - Aw_h\|_{\mathbb{V}}^2 := F(\hat{\rho}_h). \quad (\text{A.24})$$

Recall the variational principle that, because $F(\hat{\rho}_h)$ is quadratic and positive everywhere, it must achieve its minimum at $F'(\hat{\rho}_h; x_h) = 0$, where

$$F'(\hat{\rho}_h; x_h) = \lim_{\alpha \rightarrow 0} \frac{F(\hat{\rho}_h + \alpha x_h) - F(\hat{\rho}_h)}{\alpha} = (y - Bw_h, -Bx_h)_{\mathbb{V}} \quad (\text{A.25})$$

is the Gateaux derivative of the Function $F(\hat{\rho}_h)$ in a direction x_h . In this case, we have the problem:

Find $u_h \in \mathbb{U}_h$ s.t.

$$\begin{aligned} (Bu_h, Bw_h)_{\mathbb{V}} &= (y, Bw_h)_{\mathbb{V}} \quad \forall w_h \in \mathbb{U}_h \\ \rightarrow (R_h^{-1}(Au_h), R_h^{-1}(Aw_h))_{\mathbb{V}} &= (R_h^{-1}(f), R_h^{-1}(Aw_h))_{\mathbb{V}} \\ \rightarrow \langle R_h^{-1}(Au_h), Aw_h \rangle_{\mathbb{V}, \mathbb{V}^*} &= \langle R_h^{-1}(f), Aw_h \rangle_{\mathbb{V}, \mathbb{V}^*} \end{aligned} \quad (\text{A.26})$$

By defining $r_h = y - Bu_h$ and taking its inner product in \mathbb{V} with a test function v_h , we obtain

$$\begin{aligned} (r_h, v_h)_{\mathbb{V}} + (R^{-1}(Au_h), v_h)_{\mathbb{V}} &= (R^{-1}(f), v_h)_{\mathbb{V}} \quad \forall v_h \in V_h \\ \rightarrow (r_h, v_h)_{\mathbb{V}} + \langle Au_h, v_h \rangle_{\mathbb{V}^*, \mathbb{V}} &= \langle f, v_h \rangle_{\mathbb{V}^*, \mathbb{V}} \quad \forall v_h \in V_h \end{aligned} \quad (\text{A.27})$$

and from the condition (A.26), we notice we require:

Find $u_h \in \mathbb{U}_h$, $r_h \in \mathbb{V}_h$ s.t.

$$\langle Aw_h, r_h \rangle_{\mathbb{V}^*, \mathbb{V}} = 0 \quad \forall w_h \in \mathbb{U}_h. \quad (\text{A.28})$$

Finally, we note (equation (2.3)) that in our setting, our pairings are equivalent to the operators $b(\cdot, \cdot)$, $l(\cdot)$. Thus, our problem is written as:

Find $u_h \in \mathbb{U}_h$, $r_h \in \mathbb{V}_h$ s.t

$$(r_h, v_h)_{\mathbb{V}} + b(u_h, v_h) = l(v_h) \quad \forall v_h \in \mathbb{V}_h$$

$$b(w_h, r_h) = 0 \quad \forall w_h \in \mathbb{U}_h. \quad \blacksquare$$

A.10 The Sufficiency of One Application of Red Refinement for DDMRes on the Peterson Mesh

We can think of a Peterson Mesh of degree n of being comprised of n ‘strips’ of width h , where each strip has the geometry of Figure 15.

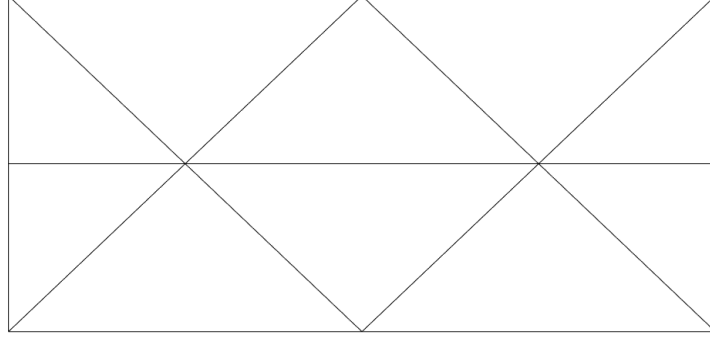


Figure 15: One of the ‘strips’ associated with the $n = 2$ Peterson mesh.

This visualisation will aid us to count the number of elements in terms of n . Looking at the image, it is clear each strip is divided into two smaller strips, each containing $2n + 1$ elements. Thus, the number of elements in the mesh is simply $N_{\text{elems}} = 2n(2n + 1) = 4n^2 + 2n$. The number of vertices is slightly more complicated. If we consider the boundary of the entire mesh, we may count $n + 1$ vertices on the top and bottom, and $2n - 1$ on each side for a total of $6n$ vertices on the boundary. Then, if we examine the strips we may see that for each strip we have n vertices in the middle, counted for all n strips. We will also need to count the number of vertices on either the top or bottom (of which there are $n - 1$) a total of $n-1$ times. So we have

$$N_{\text{verts}} = 6n + n^2 + (n - 1)(n - 1) = 2n^2 + 4n + 1. \quad (\text{A.29})$$

Note it is easy to see at this point that $N_{\text{elems}} > N_{\text{verts}} \quad \forall n > 1$, which is the inequality we wish to reverse. Now, for the red refinement to be sufficient, we require that it adds a number of vertices $N_{\text{new_verts}}$ such that

$$N_{\text{new_verts}} \geq 4n^2 + 2n - (2n^2 + 4n + 1) \quad (\text{A.30})$$

$$. \geq 2n^2 - 2n - 1 \quad (\text{A.31})$$

Counting the number of new vertices introduced during the red refinement requires only the observation that every line is bisected for each triangle. So, $N_{\text{new_verts}}$ is simply the number of lines that connect the old vertices in the first mesh. To count these, we will again look to our strip construction. The number of new vertices associated with the border is again $6n$, and looking to each strip we notice that we can associate each of the $4n + 2$ elements with one of the interior lines in a circular way (Figure 16).

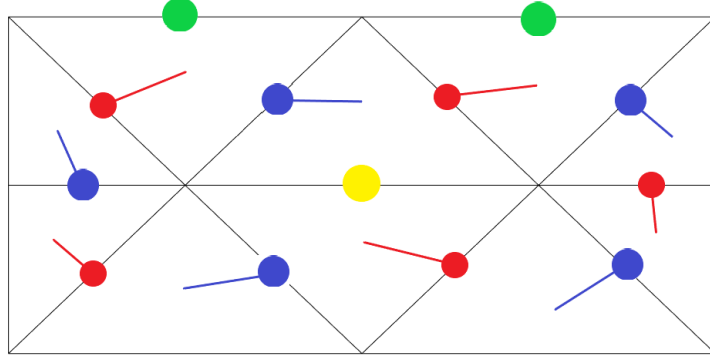


Figure 16: Blue and red vertices are associated with the element directly ‘clockwise’ of them. Yellow vertices must be counted n times, and green vertices must be counted $n - 1$ times.

So in total we can see that

$$\begin{aligned} N_{\text{new_verts}} &= 6n + n((4n + 2) + (n - 1)) + (n - 1)n \\ &= 6n + 4n^2 + 2n + n^2 - n + n^2 - n \end{aligned} \tag{A.32}$$

$$= 6n^2 + 6n \tag{A.33}$$

which clearly satisfies (A.30). However, due to the boundary conditions imposed on \mathbb{V}_h , we will not be including all the border vertices in the resultant system. To be certain we have enough for all cases, we must remove all of the border vertices (of which there are now $12n$), and we will see that we have a number of ‘surplus’ vertices equivalent to

$$\begin{aligned} N_{\text{surplus}} &= 6n^2 + 6n - 2n^2 + 2n + 1 - 12n \\ &= 4n^2 - 4n + 1. \\ &> 0 \quad \forall n \in \mathbb{N}. \end{aligned}$$

■

A.11 Reduction to Petrov-Galerkin for the Vertical Refinement Scheme

Similarly to A.10, we will need to perform a vertical refinement scheme for general n , count the number of new vertices introduced and recall that we have a Petrov-Galerkin Scheme in the case where $N_{\text{surplus}} = 0$. Quite simply, we count the number of vertices in the vertical refinement scheme by splitting the new lines into ‘odd’ (drawn at odd multiples of $\frac{h}{2}$) and ‘even’ (drawn at even multiples of $\frac{h}{2}$). It is easy to see that there are n odd strips introducing $n + 1$ new vertices each, and $n - 1$ even strips introducing n new vertices each. Thus, our surplus (before the exclusion of outflow vertices) is

$$\begin{aligned}
 N_{\text{surplus}} &= n(n + 1) + (n - 1)n - 2n^2 + 2n + 1 && \text{(from eqn A.64)} \\
 &= n^2 + n + n^2 - n - 2n^2 + 2n + 1 \\
 &= 2n + 1. && \text{(A.34)}
 \end{aligned}$$

We also note that in this particular refinement, each single edge has precisely $2n + 1$ vertices associated with it. As a result, any system with a flow perpendicular to one of the sides will exclude precisely $2n + 1$ vertices, and our system will be square. As a result, we will have $\dim(\mathbb{U}_h) = \dim(\mathbb{V}_h)$ and our method is reduced to a Petrov-Galerkin Scheme as described in 4.2.2. ■

B References

- [1] I. Muga, M.J.W. Tyler and K. G. van der Zee, *The Discrete-Dual Minimal-Residual Method (DDMRes) for Weak Advection-Reaction Problems in Banach Spaces*, De Gruyter, *Compt. Methods Appl. Math.* 2019; 19(3):557-559. <https://doi.org/10.1515/cmam-2018-0199>.
- [2] L. Chen, *Inf-Sup Conditions For Operator Equations*. <https://www.math.uci.edu/~chenlong/226/inf-supOperator.pdf>, visited 03 September 2020.
- [3] N. Saito, *Notes on the Banach-Nečas-Babuška Theorem and Kato's Minimum Modulus of Operators*. <https://arxiv.org/pdf/1711.01533.pdf>
- [4] A. Ern and J.-L. Guermond, *Theory and Practice of Finite Elements*, *Appl. Math. Sci.* 159, Springer, 2004. ISBN 978-1-4757-4355-5
- [5] M.G Larson, F. Bengzon, *The Finite Element Method: Theory, Implementation and Applications*, Springer, 2013. ISBN 978-3-642-33287-6
- [6] C. Johnson, *Numerical Solution of Partial Differential Equations by the Finite Element Method*, Cambridge Univ. Press, 1988. ISBN 0-521-345-146
- [7] G. Houzeaux, B. Eguzkitza, M. Vázquez, *A Variational Multiscale Model for the Advection-Diffusion-Reaction Equation*, *Commun. Numer. Meth. Engng* 2009; 25:787–809. <https://doi.org/10.1002/cnm.1156>
- [8] L. Demkowicz, J. Gopalakrishnan, *A Class of Discontinuous Petrov-Galerkin Methods. Part I: The Transport Equation* (2010). Portland State Univ. Mathematics and Statistics Faculty Publications and Presentations. 55. <http://archives.pdx.edu/ds/psu/10685>
- [9] D.A. Di Pietro, A. Ern, *Mathematical Aspects of Discontinuous Galerkin Methods*, Springer, 2012. ISBN 978-3-642-22979-4

- [10] T.E. Peterson *A Note on the Convergence of the Discontinuous Galerkin Method for a Scalar Hyperbolic Equation* SIAM Journal on Numerical Analysis , Feb., 1991, Vol. 28, No. 1 (Feb., 1991), pp. 133-140. <http://www.jstor.com/stable/2157936>
- [11] Website: <https://blackpawn.com/texts/pointinpoly/>