The 8th European Congress on Computational Methods in Applied Sciences and Engineering
ECCOMAS Congress 2022
5-–9 June 2022, Oslo, Norway

PETROV-GALERKIN FORMULATION EQUIVALLENT TO THE RESIDUAL MINIMIZATION METHOD FOR FINDING AN OPTIMAL TEST FUNCTIONS

MACIEJ R. PASZYŃSKI[1], TOMASZ D. ŚLUZALEC[2]

[1] AGH University of Science and Technology
Al. Mickiewicza 30, 30-059 Kraków, Poland
e-mail: maciej.paszynski@agh.edu.pl, home.agh.edu.pl/paszynsk

[2] Jagiellonian University
Gołebia 24, 31-007 Kraków, Poland
email: tomasz.sluzalec@gmail.com

Key words: Petrov-Galerkin method, optimal test functions, Deep Neural Networks

Abstract. Numerical solutions of Partial Differential Equations with Finite Element Method have multiple applications in science and engineering. Several challenging problems require special stabilization methods to deliver accurate results of the numerical simulations. The advection-dominated diffusion problem is an example of such problems. They are employed to model pollution propagation in the atmosphere. Unstable numerical methods generate unphysical oscillations, and they make no physical sense. Obtaining accurate and stable numerical simulations is difficult, and the method of stabilization depends on the parameters of the partial differential equations. They require a deep knowledge of an expert in the field of numerical analysis. We propose a method to construct and train an artificial expert in stabilizing numerical simulations based on partial differential equations. We create a neural network-driven artificial intelligence that makes decisions about the method of stabilizing computer simulations. It will automatically stabilize difficult numerical simulations in a linear computational cost by generating the optimal test functions. These test functions can be utilized for building an unconditionally stable system of linear equations. The optimal test functions proposed by artificial intelligence will not depend on the right-hand side, and thus they may be utilized in a large class of PDE-based simulations with different forcing and boundary conditions. We test our method on the model one-dimensional advection-dominated diffusion problem.

1 INTRODUCTION

The finite element method utilizes high-order basis functions, e.g., Lagrange polynomials in the classical finite element method (FEM) [5] or B-spline basis functions in isogeometric analysis (IGA) [2]. There are several challenging problems solved by FEM and IGA, such as analysis of the construction of civil engineering structures, cars or airplanes [7], geophysical applications like identification of oil and gas bearing formations [12], bioengineering simulations like modeling of cancer growth [8], blood flow simulations [14], wind turbine aerodynamics [4] or modeling of propagation of acoustic and electromagnetic waves over the human head [3]. They require

special stabilization methods to deliver high accuracy numerical solution.

The Deep Neural Networks (DNN) can be modeled as a sequence of layers, represented by linear operators and non-linear activation functions between them

$$y_{out} = ANN(x_{in}) = \\ \theta_n \sigma(...\sigma(\theta_2 \sigma(\theta_1 x_{in} + \phi_1) + \phi_2) + ...) + \phi_n \tag{1}$$

where $\{\theta_j\}_{j=1...n}$ are matrices of different sizes, and $\{\phi_j\}_{j=1...n}$ are different length vectors, and both vectors and matrices coefficients are obtained by the training procedure. The sparsity and the number of the matrices depends on the selected deep neural network kind. There are several choices possible for the non-linear activation function (sigmoid, rectified linear unit (ReLU) or leaky ReLU [1, 15]), and the classical choice is the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$. In the end, the result of the approximation with the DNN is a composition of the activation functions and linear operators.

We plan to train the neural network to find the optimal test functions stabilizing the PDE solvers. The simulations of difficult, unstable time-dependent problems, like advection-dominated diffusion or Navier-Stokes problems [9, 10] have several important applications in science and engineering. There are several stabilization methods, such as Streamline-Petrov Upwind Galerkin method (SUPG) [6], discontinuous Galerkin method (DG) [11, 13], as well as residual minimization (RM) method [9, 10].

We will use the Petrov-Galerkin formulation with the optimal test functions. It can be derived directly from the RM method. The RM for a given trial space it uses the larger test space, while for the Petrov-Galerkin formulation, we can compute the optimal test functions living in the subspace of the test space. The number of the optimal test functions is equal to the dimension of the trial space. The Petrov-Galerkin formulation, used for stabilization, enables interfacing with DNN. The DNN can be trained by running several simulations and using the computations of the optimal test functions. The test functions will be parameterized using the B-spline basis. The input to the DNN will be the problem parameters and the trial space. The output from the DNN will be the optimal test functions coefficients. Later, by running the simulations in every time step, using the actual configuration of parameters from the current time step, we can ask the DNN to provide the optimal test functions that will stabilize the computations for a given trial space.

2 EXEMPLARY ONE-DIMENSIONAL ADVECTION-DOMINATED DIFFUSION PROBLEM

Let us introduce a one-dimensional advection-dominated diffusion problem

$$\epsilon u''(x) - u'(x) = -1 \quad x \in [0,1]; \quad u(0) = u(1) = 0 \tag{2}$$

The exact solution to this problem is

$$y(x) = ((e^{1/\epsilon} - 1)x + 1 - e^{x/\epsilon})/(e^{1/\epsilon} - 1) \tag{3}$$

The problem in a weak form is given by:

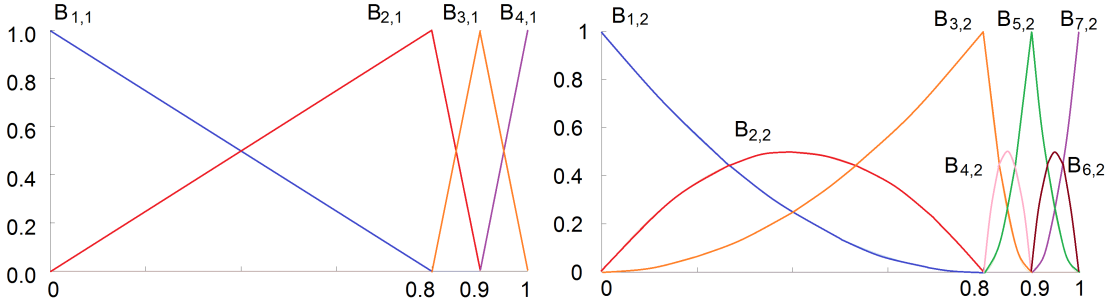

Figure 1: Space of linear B-splines, and space of quadratic B-splines defined over elements [0,0.8], [0.8,0.9] and [0.9,1].

find $u \in U = H_0^1(0,1)$ such that

$$b(u,v) = l(v) \quad \forall v \in V = H_0^1(0,1) \tag{4}$$

$$b(u,v) = \epsilon \left(u'(x), v'(x)\right)_{L^2(0,1)} + \left(u'(x), v(x)\right)_{L^2(0,1)} =$$

$$\epsilon \int_0^1 u'(x), v'(x) dx + \int_0^1 u'(x), v(x) dx \tag{5}$$

$$l(v) = (1, v(x))_{L^2(0,1)} = \int_0^1 v(x) dx \tag{6}$$

When the parameter ϵ gets small, the problem becomes numerically unstable.

To illustrate the stabilization issue, we will start with a mesh already refined towards the right endpoint, the gradient of the solution is large.

Let us introduce the knot vector [0 0 0.8 0.9 1 1] defining four linear B-splines over three elements $[0, 0.8]$, $[0.8, 0.9]$ (how to translate the knot vector into B-splines is defined here https://epodreczniki.open.agh.edu.pl/handbook/1088/module/ 1098/reader and in the next sections there) and $[0.9, 1]$ defining the trial space $U_h = span\{B_{1,1}, B_{2,1}, B_{3,1}, B_{4,1}\}$ as presented in left-panel in Figure 1.

Let us also introduce the knot vector [0 0 0 0.8 0.8 0.9 0.9 1 1 1] defining seven quadratic B-spline basis functions with C^0 separators spanning the test space presented in right-panel in Figure 1, $V_h = span\{B_{1,2}, B_{2,2}, B_{3,2}, B_{4,2}, B_{5,2}, B_{6,2}, B_{7,2}\}$.

3 GALERKIN FORMULATION

The discrete Galerkin form is given by: find $u_h(x) = \sum_{i=1,...,7} u_i B_{i,1}$ (quadratic B-splines) such that

$$b(u_h, v_h) = l(v_h)$$

$$\forall v_h \in V_h = span\{B_{1,2}, B_{2,2}, B_{3,2}, B_{4,2}, B_{5,2}, B_{6,2}, B_{7,2}\} \tag{7}$$

We can also enforce zero Dirichlet b.c. be setting the first and the last row in the system, corresponding to the first and the last trial function, to zero, with 1 on diagonal and 0 on the right-hand side. The Galerkin formulation:

$$Au = F \tag{8}$$

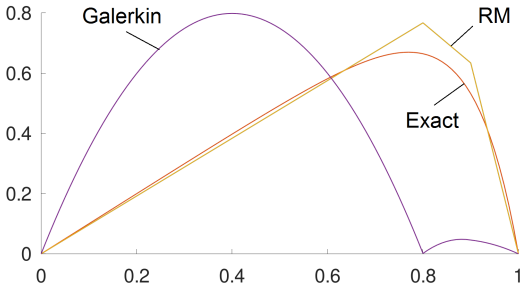

Figure 2: Comparison of Galerkin method with trial=test=quadratic B-splines with C^0 separators (defined by knot vector [0 0 0 0.8 0.8 0.9 0.9 1 1 1]), the residual minimization with linear B-splines for trial (defined by knot vector [0 0 0.8 0.9 1 1]) and quadratic B-splines with C^0 separators for test (defined by knot vector [0 0 0 0.8 0.8 0.9 0.9 1 1 1]) for $\epsilon = 0.1$, and the exact solution for $\epsilon = 0.1$.

$$A = \begin{bmatrix} 1.0 & 0.0 & \cdots & 0.0 \\ b(B_{1,1}, B_{2,2}) & b(B_{2,1}, B_{2,2}) & \cdots & b(B_{7,1}, B_{2,2}) \\ \cdots & \cdots & \cdots & \cdots \\ b(B_{1,1}, B_{6,2}) & b(B_{2,1}, B_{6,2}) & \cdots & b(B_{7,1}, B_{6,2}) \\ 0.0 & 0.0 & \cdots & 1.0 \end{bmatrix} u = \begin{bmatrix} u_1 \\ u_2 \\ \cdots \\ u_6 \\ u_7 \end{bmatrix} \quad F = - \begin{bmatrix} l(B_{1,2}) \\ l(B_{2,2}) \\ \cdots \\ l(B_{7,2}) \end{bmatrix} \tag{9}$$

where by red color we denote basis functions from the test space, and by blue color we denote basis functions from the trial space, and trial = test = quadratic B-splines with C0 sepparators.

The Galerkin formulation of the standard finite element method with the space of quadratic B-splines for small $\epsilon = 0.1$ parameter results in some oscillations and large numerical error see Figure 2. To obtain a high quality numerical solution to this problem, a special stabilization method is required.

4 PETROV-GALERKIN FORMULATION

The discrete Petrov-Galerkin form is given by: find $u_h(x) = \sum_{i=1,\ldots,4} u_i B_{i,1}$ such that

$$\begin{aligned} b(u_h, v_h) &= l(v_h) \\ \forall v_h \in V_h = span\{B_{1,2}, B_{2,2}, B_{3,2}, B_{4,2}, B_{5,2}, B_{6,2}, B_{7,2}\} \end{aligned} \tag{10}$$

$$B = \begin{bmatrix} b(B_{1,1}, B_{1,2}) & b(B_{2,1}, B_{1,2}) & \cdots & b(B_{4,1}, B_{1,2}) \\ b(B_{1,1}, B_{2,2}) & b(B_{2,1}, B_{2,2}) & \cdots & b(B_{4,1}, B_{2,2}) \\ \cdots & \cdots & \cdots & \cdots \\ b(B_{1,1}, B_{7,2}) & b(B_{2,1}, B_{7,2}) & \cdots & b(B_{4,1}, B_{7,2}) \end{bmatrix} u = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} \quad F = \begin{bmatrix} l(B_{1,2}) \\ l(B_{2,2}) \\ \cdots \\ l(B_{4,2}) \end{bmatrix} \tag{11}$$

The Petrov-Galerkin formulation here (with different trial and test spaces) requires some transformation, since we have different number of unknowns and equations, and it is not possible to solve in this form.

5 RESIDUAL MINIMIZATION FORMULATION

The residual minimization method requires:

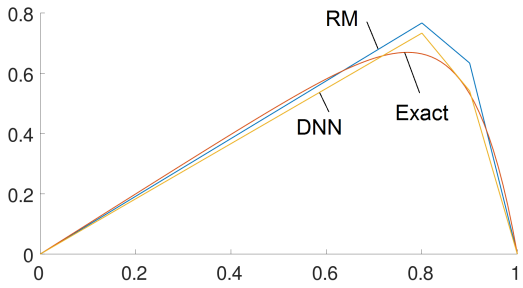

Figure 3: Comparison of the solution computed with the optimal test functions found by the solver, the solution computed with quasi-optimal test functions found by the artificial neural network, and the exact solution for $\epsilon = 0.1$.

- Selection of the trial space.
- Selection of the test space.
- Selection of the inner product for minimization of the residual, that will form the Gram matrix.

In our simple representative example, we will take smaller trial space (approximating the solution) U_h defined with linear B-splines, and larger test space, namely the quadratic B-splines with C^0 separators. In the residual minimization method, we also prescribe the norm (and the scalar product) that is supposed to minimize the residual of the solution (minimize the numerical error). We select the H^1 scalar product $g(u,v) = \int_0^1 (uv + u'v')dx$, so the residual minimization stabilization of (10) is

$$\begin{bmatrix} G & -B \\ B^T & 0 \end{bmatrix} \begin{bmatrix} r \\ u \end{bmatrix} = \begin{bmatrix} F \\ 0 \end{bmatrix} \tag{12}$$

where G stands for the Gram matrix, expressing the scalar product selected for the residual minimization procedure

$$G = \begin{bmatrix} g(B_{1,2},B_{1,2}) & g(B_{1,2},B_{2,2}) & \cdots & g(B_{1,2},B_{7,2}) \\ g(B_{2,2},B_{1,2}) & g(B_{2,2},B_{2,2}) & \cdots & g(B_{2,2},B_{7,2}) \\ \cdots & \cdots & \cdots & \cdots \\ g(B_{7,2},B_{1,2}) & g(B_{7,2},B_{2,2}) & \cdots & g(B_{7,2},B_{7,2}) \end{bmatrix}, \tag{13}$$

$$B = \begin{bmatrix} b(B_{1,1},B_{1,2}) & b(B_{2,1},B_{1,2}) & \cdots & b(B_{4,1},B_{1,2}) \\ b(B_{1,1},B_{2,2}) & b(B_{2,1},B_{2,2}) & \cdots & b(B_{4,1},B_{2,2}) \\ \cdots & \cdots & \cdots & \cdots \\ b(B_{1,1},B_{7,2}) & b(B_{2,1},B_{7,2}) & \cdots & b(B_{4,1},B_{7,2}) \end{bmatrix} u = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} \quad r = \begin{bmatrix} r_1 \\ r_2 \\ \cdots \\ r_7 \end{bmatrix} \quad F = - \begin{bmatrix} l(B_{1,2}) \\ l(B_{2,2}) \\ \cdots \\ l(B_{7,2}) \end{bmatrix}$$

where by red color we denote basis functions from the test space, and by blue color we denote basis functions from the trial space. Here (u_1, u_2, u_3, u_4) is the solution, and $(r_1, r_2, r_3, r_4, r_5, r_6, r_7)$ stands for the residual (representing the local error).

6 OPTIMAL TEST FUNCTIONS

This solution with the residual minimization (RM) method is equivalent to the Petrov-Galerkin (PG) formulation with the optimal test functions. We can either solve RM formulation, or compute the optimal test functions and solve PG formulation with the optimal test functions.

W can compute the optimal test functions to provide the results equivalent to the residual minimization method with a given trial and test spaces. We will later train an artificial neural network to tell us the formulas for the optimal test functions.

For the basis of the trial space $\{z^1, z^2, z^3, z^4\} = \{ \begin{bmatrix}1\\0\\0\\0\end{bmatrix} \begin{bmatrix}0\\1\\0\\0\end{bmatrix} \begin{bmatrix}0\\0\\1\\0\end{bmatrix} \begin{bmatrix}0\\0\\0\\1\end{bmatrix} \}$ the optimal test functions $\{w^1, w^2, w^3, w^4\}$ are obtained by $w^k = G^{-1}Bz^k$. where G is the Gram matrix, and B is the matrix correspondig to the weak form (10). We have to solve the system of linear equations for each optimal test function

$$Gw^k = Bz^k \tag{14}$$

These optimal test functions $V_h^{opt} = span\{w^1, w^2, w^3, w^4\}$ they form a subspace $V_h^{opt} \subset V_h$ such that the Petrov-Galerkin formulation with the optimal test functions gives the best possible (up to the trial space used) solution to the advection-dominated diffusion problem. Namely, we get this solution by solving

$$\begin{bmatrix} b(B_{1,1},w^1) & b(B_{2,1},w^1) & b(B_{3,1},w^1) & b(B_{4,1},w^1) \\ b(B_{1,1},w^2) & b(B_{2,1},w^2) & b(B_{3,1},w^2) & b(B_{4,1},w^2) \\ b(B_{1,1},w^3) & b(B_{2,1},w^3) & b(B_{3,1},w^3) & b(B_{4,1},w^3) \\ b(B_{1,1},w^4) & b(B_{2,1},w^4) & b(B_{3,1},w^4) & b(B_{4,1},w^4) \end{bmatrix} \begin{bmatrix} u_1^{opt} \\ u_2^{opt} \\ u_3^{opt} \\ u_4^{opt} \end{bmatrix} = \begin{bmatrix} l(w^1) \\ l(w^2) \\ l(w^3) \\ l(w^4) \end{bmatrix} \tag{15}$$

where $(u_1^{opt}, u_2^{opt}, u_3^{opt}, u_4^{opt})$ is the optimal solution.

This formulation gives a possibility of finding the optimal test functions for a given PDE parameters (in our case ϵ) and the given trial space U_h. *We can employ the artificial neural network (ANN) and train it to provide the optimal test functions for given ϵ and for a given trial space. These optimal test functions are independent on the right-hand side and boundary condition, thus the ANN can be applied for stabilization of a large class of problems.*

7 ARTIFICIAL NEURAL NETWORK FINDING THE OPTIMAL TEST FUNCTIONS

All tables should be numbered consecutively and captioned, theIn this section, we illustrate on a very simple example the idea of constructing and training the artificial neural network (ANN) finding the optimal test functions for stabilization of our exemplary model advection-dominated diffusion problem.

For given trial and test spaces, let us introduce the family of simple artificial neural network (ANN)

$$ANN_i^k(\epsilon) = w_i^k \tag{16}$$

Given ϵ parameter it returns the optimal test functions coordinate w_i^k $k = 1, 2, 3, 4$, $i = 1, ..., 7$. In this simple example we create one single layer ANN for each coefficient. In the more complicated

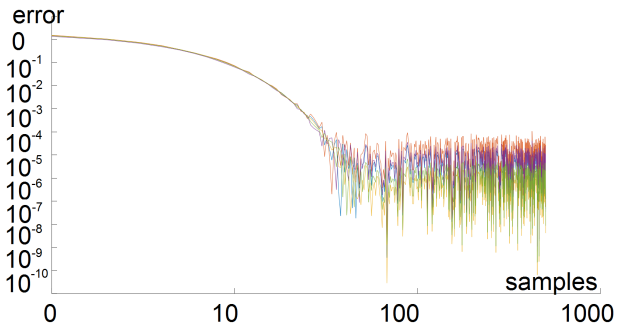

Figure 4: Convergence of the ANN training for a single weight (convergence of $a_i^k, b_i^k, c_i^k, d_i^k$ approximating $w_i^k = ANN(\epsilon) = c_i^k \sigma\left(a_i^k \epsilon + b_i^k\right) + d_i^k$).

cases, we should construct one DNN for the entire problem, and the input and output of the DNN will depend on the model parameters and the trial and test spaces assumed. We create a simple one-layer neural network,

$$ANN_i^k(n) = c_i^k \sigma\left(a_i^k \epsilon + b_i^k\right) + d_i^k = w_i^k \tag{17}$$

for each coefficients w_i^k, where the activation function is sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{18}$$

The goal of the training is to find values of the weights $a_i^k, b_i^k, c_i^k, d_i^k$. For selected trial U_h and test V_h spaces, as presented in Figure 1, we prepare a set of samples

- We randomly select $\epsilon \in (0,1)$
- We find the optimal test functions coefficients w_i^k by solving (14) (removing the first and last rows and columns due to zero Dirichlet b.c.)

$$\begin{bmatrix} g(B_{2,2},B_{2,2}) & \cdots & g(B_{2,2},B_{6,2}) \\ g(B_{3,2},B_{2,2}) & \cdots & g(B_{3,2},B_{6,2}) \\ \cdots & \cdots & \cdots \\ g(B_{6,2},B_{2,2}) & \cdots & g(B_{6,2},B_{6,2}) \end{bmatrix} \begin{bmatrix} w_2^k \\ w_3^k \\ \cdots \\ w_6^k \end{bmatrix} = \begin{bmatrix} b(B_{2,1},B_{2,2}) & b(B_{3,1},B_{2,2}) \\ b(B_{2,1},B_{3,2}) & b(B_{3,1},B_{3,2}) \\ \cdots & \\ b(B_{2,1},B_{6,2}) & b(B_{3,1},B_{6,2}) \end{bmatrix} \begin{bmatrix} \delta_{1k} \\ \delta_{2k} \end{bmatrix} \tag{19}$$

In other words, we train the neural network based on ϵ parameter samples to find the optimal test functions. How to train the artificial neural network? We define the error function

$$e_i^k(\epsilon) = 0.5\left(ANN_i^k(\epsilon) - w_i^k(\epsilon)\right)^2 = 0.5\left(c_i^k \sigma\left(a_i^k \epsilon + b_i^k\right) + d_i^k - w_i^k(\epsilon)\right)^2 \tag{20}$$

Now, we compute the derivatives

$$\frac{\partial e_i^k(\epsilon)}{\partial a_i^k} = \frac{c_i^k \epsilon \; exp(-a_i^k \epsilon - b_i^k)(ANN_i^k(\epsilon) - w_i^k(\epsilon))}{(exp(-a_i^k \epsilon - b_i^k)+1)^2}$$
$$\frac{\partial e_i^k(\epsilon)}{\partial b_i^k} = \frac{c_i^k exp(-a_i^k \epsilon - b_i^k)(ANN_i^k(\epsilon) - w_i^k(\epsilon))}{(exp(-a_i^k \epsilon - b_i^k)+1)^2}$$
$$\frac{\partial e_i^k(\epsilon)}{\partial c_i^k} = \frac{(ANN_i^k(\epsilon) - w_i^k(\epsilon))}{(exp(-a_i^k \epsilon - b_i^k)+1)}$$
$$\frac{\partial e_i^k(\epsilon)}{\partial d_i^k} = (ANN_i^k(\epsilon) - w_i^k(\epsilon)) \tag{21}$$

which say "how fast the error is changing if I modify a given coefficient". We loop through the data set $\{\epsilon, (w_i^k(\epsilon))\}_{\epsilon \in A}$ where A is the set of selected points from $(0,1)$, and we train each of the ANN_i^k.

1. Select $(\epsilon, (w_i^k))$
2. Compute $w_i^k = ANN_i^k(\epsilon) = c_i^k \sigma\left(a_i^k \epsilon + b_i^k\right) + d_i^k$, compute $e_i^k(\epsilon)$, and $\frac{\partial e_i^k(\epsilon)}{\partial a_i^k}$, $\frac{\partial e_i^k(\epsilon)}{\partial b_i^k}$, $\frac{\partial e_i^k(\epsilon)}{\partial c_i^k}$, $\frac{\partial e_i^k(\epsilon)}{\partial d_i^k}$
3. Correct $a_i^k = a_i^k - \eta \frac{\partial e_i^k(\epsilon)}{\partial a_i^k}$, $b_i^k = b_i^k - \eta * \frac{\partial e_i^k(\epsilon)}{\partial b_i^k}$, $c_i^k = c_i^k - \eta * \frac{\partial e_i^k(\epsilon)}{\partial c_i^k}$, $d_i^k = d_i^k - \eta * \frac{\partial e_i^k(\epsilon)}{\partial d_i^k}$, $\eta \in (0,1)$. This is like one step of a local gradient method.

The training for a single weight can be implemented in MATLAB function, called for data generated in the training set $(dataset\_in(i), dataset\_w(k,i))$, generated for a weight w_n^m of the optimal test function, where where $k = 5*(m-1)+n$ and i represents the dataset index.

```
% Training of ANN approximating a single weight
a(k)=1.0; b(k)=1.0; c(k)=1.0; d(k)=1.0; eta(k)=0.1;
for i=1:ndataset
eval(k) = c(k)* 1.0/(1.0+exp(-(a(k)*DS_in(i)+b(k))))+d(k);
error(k) = 0.5*(eval(k)-DS_w(k,i))^2;
derrorda = c(k)*DS_in(i)*exp(-a(k)*DS_in(i)-b(k))*(eval(k)-DS_w(k,i))/
(exp(-a(k)*DS_in(i)-b(k))+1)^2; a(k)-=eta(k)* derrorda;
derrordb = c(k)*exp(-a(k)*DS_in(i)-b(k))*(eval(k)-DS_w(k,i))/(exp(-a(k)*DS_in(i)-b(k));
b(k)-=eta(k)* derrordb;
derrordc = (eval(k)-DS_w(k,i))/(exp(-a(k)*DS_in(i)-b(k))+1); c(k)-=eta(k)*derrordc;
derrordd = (eval(k)-DS_w(k,i));d(k)-=eta(k)*derrordd;
```

The weight value as approximated by the trained ANN can be computed using the following procedure

```
% evaluation of ANN approximation of weights for a given ϵ
for m=2,3
for n=2,6
k=5*(m-1)+n;eval(k)=c(k)*1.0/(1.0+exp(-(a(k)*epsilon+b(k))))+d(k);
```

Due to zero Dirichlet b.c., the optimal test functions are given by the knot vector [0 0 0 0.8 0.8 0.9 0.9 1 1 1] and weights [0 eval(1) eval(2) eval(3) eval(4) eval(5) 0] and [0 eval(6) eval(7) eval(8) eval(9) eval(10) 0].

The convergence of the training procedure for a single weight (the errors of evaluating a_i^k, b_i^k, c_i^k, and d_i^k) are presented in right panel in Figure 3. Once we have the optimal test functions, we can use them to compute the solution from the Petrov-Galerkin formulation (15). They give quite similar results, presented in Figure 3. To improve the quality of the optimal test functions and the quality of the numerical result, we can replace the several simple one layered ANN trained one for each parameter by more complex DNN.

8 CONCLUSIONS

- We presented a method for stabilization of a class of advection-diffusion problems with different right-hand sides.
- Our method precomputes the optimal test functions for the Petrov-Galerkin formulation using artificial neural networks.
- We tested our method on a model advection-dominated problem in one dimension.
- Future work will involve generalization of the method into two- or three-dimensional problems, including advection-dominated diffusion and Navier-Stokes equations.
- Future work will also involve the development of fast solvers for the stabilized Petrov-Galerkin formulations.

9 APPENDIX A: RESIDUAL MINIMIZATION METHOD

We employ the residual minimization method for stabilization. It can be derived as follows. For a general weak problem: Find $u \in U$ such as

$$b(u,v) = l(v) \quad \forall v \in V \tag{22}$$

we define the operator $B : U \to V'$ such as $\langle Bu, v\rangle_{V' \times V} = b\,(u,v)$. so we can reformulate the problem as

$$Bu - l = 0 \tag{23}$$

We wish to minimize the residual

$$u_h =_{w_h \in U_h} \frac{1}{2}\|Bw_h - l\|^2_{V'} \tag{24}$$

We introduce the Riesz operator $R_V \colon V \ni v \to (v,.) \in V'$ being the isometric isomorphism to project the problem back to V

$$u_h =_{w_h \in U_h} \frac{1}{2}\|R_V^{-1}(Bw_h - l)\|^2_V \tag{25}$$

The minimum is attained at u_h when the Gâteaux derivative is equal to 0 in all directions

$$(R_V^{-1}(Bu_h - l), R_V^{-1}(B\,w_h))_V = 0 \quad \forall\, w_h \in U_h \tag{26}$$

We define the residual $r = R_V^{-1}(Bu_h - l)$ and we get

$$(r, R_V^{-1}(B\, w_h)) = 0 \quad \forall\, w_h \in U_h \tag{27}$$

From the definition of R_V for all functionals $f \in V'$

$$(v, R_V^{-1} f)_V = \langle f, v\rangle \ (= f(v) \text{ from definition of } \langle\cdot,\cdot\rangle) \tag{28}$$

so in particular for $f = Bw_h$ and $v = r$ we get

$$\langle Bw_h, r\rangle = 0 \qquad \forall w_h \in U_h \tag{29}$$

From the definition of the residual we have

$$(r, v)_V = \langle Bu_h - l, v\rangle, \quad \forall v \in V. \tag{30}$$

Thus, from (29) and (30), our problem reduces to the following semi-infinite problem: Find $(r, u_h) \in V \times U_h$ such as

$$\begin{array}{lll} (r, v)_V & -\langle Bu_h, v\rangle = -\langle l, v\rangle, & \forall v \in V, \\ \langle Bw_h, r\rangle & = 0, & \forall w_h \in U_h. \end{array} \tag{31}$$

We discretize the test space $V_h \in V$ to get the discrete problem: Find $(r_h, u_h) \in V_h \times U_h$ such as

$$\begin{array}{lll} (r_h, v_h)_{V_h} & -\langle Bu_h, v_h\rangle = -\langle l, v_h\rangle, & \forall v \in V_h, \\ \langle Bw_h, r_h\rangle & = 0, & \forall w_h \in U_h, \end{array} \tag{32}$$

where $(\cdot,\cdot)_{V_h}$ is an inner product in V_h, $\langle Bu_h, v_h\rangle = b\,(u_h, v_h)$, and $\langle Bw_h, r_h\rangle = b\,(w_h, r_h)$. We select the discrete test space V_h large enough to present the optimal test functions' subspace. In the residual minimization method, we gain stability by enriching the test space V_h without changing the trial space U_h.

10 APPENDIX B: EQUIVALENCE OF THE RESIDUAL MINIMIZATION METHOD AND PETROV-GALERKIN FORMULATION

Having the residual minimization problem

$$\begin{array}{lll} (r_h, v_h)_{V_h} & -\langle Bu_h, v_h\rangle = -\langle l, v_h\rangle, & \forall v \in V_h, \\ \langle Bw_h, r_h\rangle & = 0, & \forall w_h \in U_h, \end{array} \tag{33}$$

with the test space V_h sufficiently large to stabilize the simulation, we can transform this formulation into Petrov-Galerkin one in the following way. We recall the Riesz operator

$$R_{V_h} : V_h \to V_h' \tag{34}$$

defined as

$$\langle R_{V_h} r_h, v_h\rangle = (r_h, v_h)_{V_h} \in V_h' \times V_h \tag{35}$$

and the residuum

$$r_h = R_{V_h}^{-1}(Bu_h - l) \tag{36}$$

We can substitute the residuum into the second equation in (33) to obtain

$$\langle Bw_h, R_{V_h}^{-1}(Bu_h - l)\rangle = 0 \quad \forall w_h \in U_h, \tag{37}$$

From the properties of the Riesz operator we have

$$\langle f, v_h\rangle = (R_{V_h}^{-1} f, v_h)_{V_h} \tag{38}$$

for $f \in V_h'$, where we can represent $f = R_{V_h} a$ for $a \in V_h$. Using this property for $f = R_{V_h} r_h$ we obtain

$$(R_{V_h}^{-1} Bw_h, R_{V_h}^{-1}(Bu_h - l))_{V_h} = 0 \quad \forall w_h \in U_h \tag{39}$$

Using this property once again, and from the symmetry of the scalar product we have

$$\langle Bu_h - l, R_{V_h}^{-1} Bw_h\rangle = 0 \quad \forall w_h \in U_h \tag{40}$$

which is equivallent to

$$\langle Bu_h, R_{V_h}^{-1} Bw_h\rangle = \langle l, R_{V_h}^{-1} Bw_h\rangle \quad \forall w_h \in U_h \tag{41}$$

and

$$b(u_h, R_{V_h}^{-1} Bw_h) = l(R_{V_h}^{-1} Bw_h) \quad \forall w_h \in U_h \tag{42}$$

which is the Petrov-Galerkin formulation with the optimal test functions.

11 ACKNOWLEDGEMENTS

The European Union's Horizon 2020 Research and Innovation Program of the Marie Skłodowska-Curie grant agreement No. 777778, MATHROCKs. Research project partly supported by program "Excellence initiative – research university" for the University of Science and Technology.

REFERENCES

[1] J. Berg and K. Nystrom. Datadriven discovery of pdes in complex datasets. *J. Comput. Phys.*, (2019) **384**:239–252.

[2] J. Austin Cottrell, T.J.R. Hughes, and Yuri Bazilevs. *Isogeometric analysis: toward integration of CAD and FEA*, (2009) John Wiley Sons.

[3] L. Demkowicz, P. Gatto, J. Kurtz, M. Paszyński, W. Rachowicz, E. Bleszyński, N. Bleszyński, M. Hamilton, C. Champlin, and Pardo D. Modeling of bone conduction of sound in the human head using *hp* finite elements i. code design and verification. *Comput. Methods Appl. Mech. Eng.*, (2011) 21- **22**:1757–1773.

[4] Ming-Chen Hsu, Ido Akkerman, and Yuri Bazilevs. High-performance computing of wind turbine aerodynamics using isogeometric analysis. *Comput. Fluids*, (2011) **49(1)**:93–100.

[5] T.J.R. Hughes and Ted Belytschko. The finite element method: Linear static and dynamic finite element analysis. *Comput.-Aided Civ. Infrastruct. Eng.*, (1989) **4(3)**:245–246

[6] T.J.R. Hughes, L. Franca, and M. Mallet. A new finite element formulation for computational fluid dynamics: Vi. convergence analysis of the generalized SUPG formulation for linear time-dependent multidimensional advective-diffusive systems. *Comput. Methods Appl. Mech. Eng.*, (1987) **63(1)**:97–112.

[7] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Comput. Methods Appl. Mech. Eng.*, (2005) **194(39)**:4135–4195.

[8] M. Łoś, A. Kłusek, M. A. Hassaan, K. Pingali, W. Dzwinel, and M. Paszyński. Parallel fast isogeometric l2 projection solver with GALOIS system for 3D tumor growth simulations. *Comput. Methods Appl. Mech. Eng.*, (2019) **343**:1–22.

[9] M. Łoś, J. Munoz-Matute, I. Muga, and M. Paszyński. Isogeometric residual minimization method (iGRM) with direction splitting for non-stationary advection–diffusion problems. *Comput. Math. Appl.*, (2019) **79(2)**:213–229.

[10] M. Łoś, J. Munoz-Matute, I. Muga, and M. Paszyński. Isogeometric residual minimization (iGRM) for time-dependent stokes and Navier-Stokes problems. *Comput. Math. Appl.*, (2021) **95(1)**:200-214.

[11] M. Łoś, S. Rojas, M. Paszyński, I. Muga, and V. Calo. Discontinuous Galerkin based isogeometric Residual Minimization for the Stokes problem (DGiRM). *J. Comput. Sci.*, (2021) **50**:101306.

[12] D. Pardo, C. Torres-Verdin, M. J. Nam, M. Paszyński, and V. Calo. Fourier series expansion in a ´ non-orthogonal system of coordinates for the simulation of 3d alternating current borehole resistivity measurements. *Comput. Methods Appl. Mech. Eng.* (2008) **45-48**:3836–3849.

[13] D. Pietro and A. Ern. *Mathematical Aspects of Discontinuous Galerkin Methods* (2011) Springer.

[14] C.A. Taylor, T.J.R. Hughes, and C. K. Zarins. Finite element modeling of blood flow in arteries. *Comput. Methods Appl. Mech. Eng.*, (1998) **158(1)**:151–196.

[15] G. Tsihrintzis, D. N. Sotiropoulos, and L. C. Jain. *Machine learning paradigms: Advances in data analytics*, (2019) Springer.