Dobbiaco, July 1-5, 2005
Mathematical Models in Life Science: Theory and Simulation

An Introduction to Finite Elements Computational Aspects

Alfio Quarteroni & Alessandro Veneziani

CMCS
Chaire de Modélisation et Calcul Scienti↓que
Ecole Polytechnique Fédérale de Lausanne (EPFL)
iacs.epfl.ch/cmcs

MOX
Dipartimento di Matematica
Politecnico di Milano
mox.polimi.it

Outline

- Numerical experiments for verifying properties of Finite Elements Method.
- Basic reference: A. Quarteroni, A. Valli, *Numerical Approximation of Partial Differential Equations*, Springer-Verlag, Heidelberg, 1996, Chapters 3,5.

A First Numerical Example: a 1D heat problem

`www1.mate.polimi.it/~calnum`

Consider a thin rod of length L whose temperature at $x = 0$ is fixed to t_0 while the other endpoint $x = L$ is thermally isolated. Assume that the rod has a cross-section with constant area equal to A and that the perimeter of A is p.
The temperature u of the rod at a generic point $x \in (0, L)$ is governed by the following boundary value problem with mixed Dirichlet-Neumann conditions

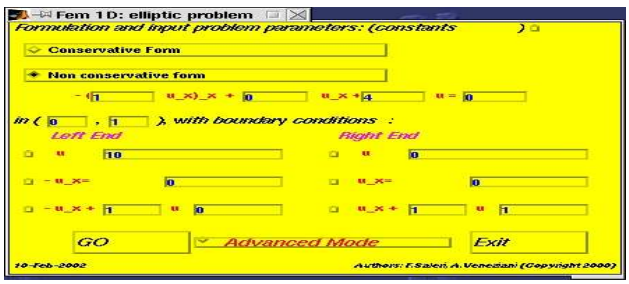
$$\begin{cases} -\mu A u'' + \sigma p u = 0 & x \in (0, L), \\ u(0) = u_0, & u'(L) = 0, \end{cases} \tag{1}$$

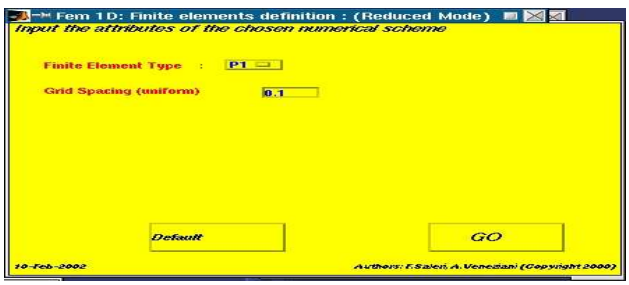
where μ denotes the thermal conductivity and σ is the convective transfer coefficient.
The exact solution of the problem is the (smooth) function

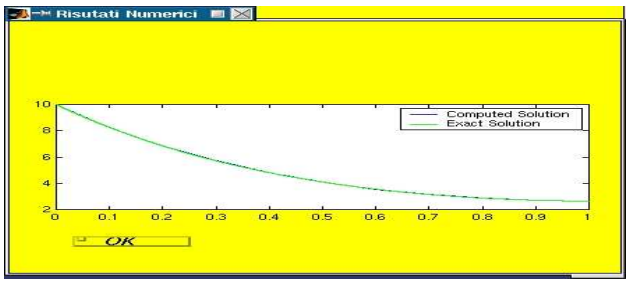
$$u(x) = u_0 \frac{\cosh[m(L-x)]}{\cosh(mL)}, \quad \text{where} \quad = \sqrt{\sigma p/\mu A}.$$

We solve the problem by using linear, quadratic and cubic finite elements on a grid with uniform size. Assume $\sigma p = 4,\ \mu A = 1,\ T_0 = 10,\ L = 1$, so that $m = 2$.

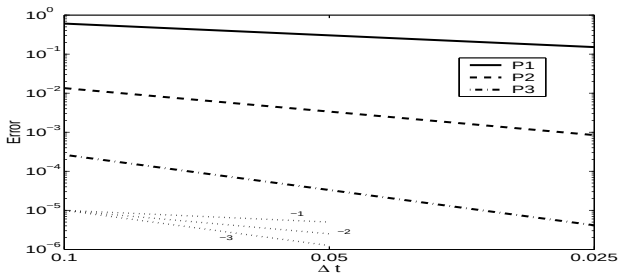
Solution with `Fem1d`







H^1 errors



- Condition number for different h and p

	$h = 0.1$	$h = 0.05$	$h = 0.025$
P1	67.35553	259.0683	1013.6147
P2	344.8292	1350.8582	5342.138
P3	1105.5703	4358.8914	17301.0288

- Iterations required by PCG with different preconditioners: $\mathtt{P}^1$

	$h = 0.1$	$h = 0.05$	$h = 0.025$
NoPreconditioner	10	20	40
Tridiag	1	1	1
ILU(0)	1	1	1

- Iterations required by `PCG` with different preconditioners: $\mathtt{P}^2$

	$h = 0.1$	$h = 0.05$	$h = 0.025$
`NoPreconditioner`	20	40	84
`Tridiag`	11	21	37
`ILU(0)`	1	1	1

- Iterations required by `PCG` with different preconditioners: $\mathtt{P}^3$

	$h = 0.1$	$h = 0.05$	$h = 0.025$
`NoPreconditioner`	32	70	145
`Tridiag`	21	49	111
`ILU(0)`	1	1	1

More on preconditioning

Let us consider a rod ($0 \leq x \leq 1$) whose vertical displacement under a vertical force $f(x)$ is denoted by $u(x)$. After some simplifications, we obtain the equation:

$$-E\frac{d^2u}{dx^2} + S\frac{d^4u}{dx^4} = f \qquad 0 < x < 1$$

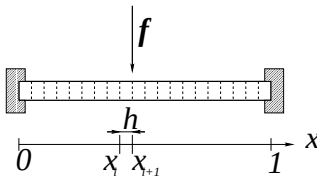
where E and S depend on the physical ad geometrical features of the rod. For a rod fixed at the endpoints, we have:

$$u = 0, \quad \frac{du}{dx} = 0 \qquad \text{for } x = 0, 1.$$

The model is obtained by some simplifications and the (linear) superimpositions of a traction term (second derivatives) and a bending one (fourth derivative).
For numerical purposes, we follow a finite different approach with uniform mesh size $\Delta x = h$:

$$E\frac{1}{h^2}\left(-u_{i+1} + 2u_i - u_{i-1}\right) + S\frac{1}{h^4}\left(u_{i+2} - 4u_{i+1} + 6u_i - 4u_{i-1} + u_{i-2}\right) = f_i, \quad (2)$$

where u_j is the displacement in $x_j = jh$, f_j stands for $f(x_j)$.



Boundary conditions can be approximated by the following relations::

$$u_0 = u(0) = 0,\ u_1 = u(h) = 0,\ u_{n-1} = u((n-1)h) = 0,\ u_n = u(nh) = u(1) = 0.$$

We obtain a linear system for the displacement in the nodes x_j, $2 \leq j \leq n-2$ ($n = 1/h$):

$$A\mathbf{u} = \mathbf{f}, \quad \text{where} \quad A = \frac{1}{h^2}T + \frac{1}{h^4}F,$$

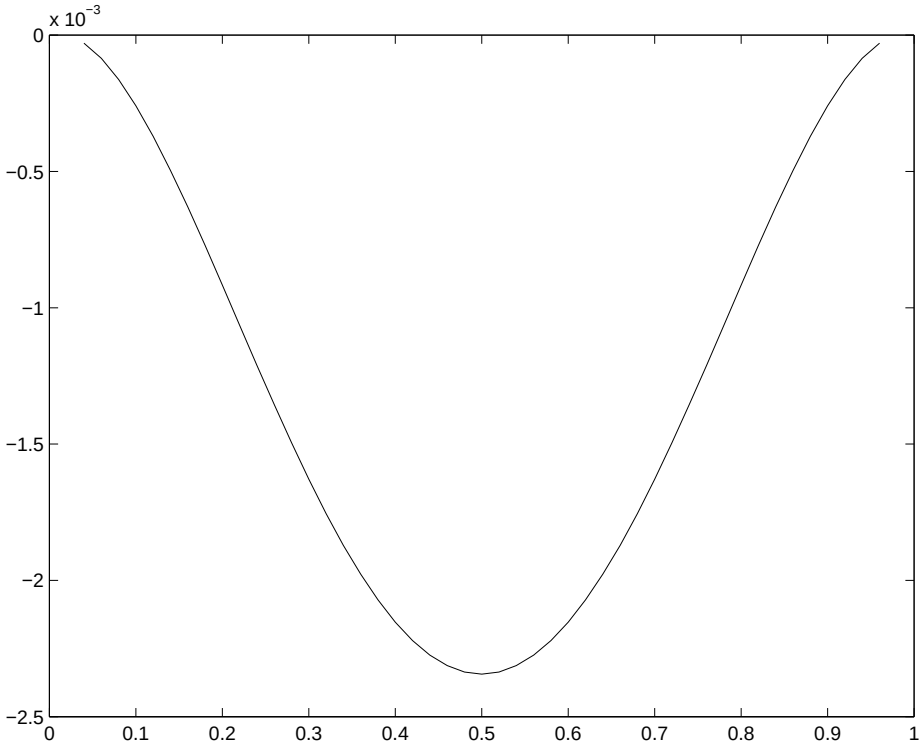
$$T = \begin{bmatrix} 2 & -1 & 0 & \dots & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & \dots & 0 & 0 \\ 0 & -1 & 2 & -1 & \dots & 0 & 0 \\ \ddots & \ddots & \ddots & \ddots & \dots & 0 & 0 \\ 0 & 0 & \dots & -1 & 2 & -1 & 0 \\ 0 & 0 & \dots & 0 & -1 & 2 & -1 \\ 0 & 0 & \dots & 0 & 0 & -1 & 2 \end{bmatrix} \quad F = \begin{bmatrix} 6 & -4 & 1 & \dots & 0 & 0 & 0 \\ -4 & 6 & -4 & 1 & \dots & 0 & 0 \\ 1 & -4 & 6 & -4 & 1 & \dots & 0 \\ \ddots & \ddots & \ddots & \ddots & \dots & \dots & 0 \\ 0 & \dots & 1 & -4 & 6 & -4 & 1 \\ 0 & 0 & \dots & 1 & -4 & 6 & -4 \\ 0 & 0 & \dots & 0 & 1 & -4 & 6 \end{bmatrix}.$$

It is easily realized that A is symmetric and positive definite, so we can solve the system with the *conjugate gradient* method.

`Matlab` instructions:

```
>> n=50; h=1/n;
>> dim=n-3;
>> T=2*eye(dim)-diag(ones(dim-1,1),1)
              -diag(ones(dim-1,1),-1);
>> F=6*eye(dim)-4*diag(ones(dim-1,1),1)
     -4*diag(ones(dim-1,1),-1)
     +diag(ones(dim-2,1),2)+
     diag(ones(dim-2,1),-2);
>> A=1/h^2*T+1/h^4*F;
>> f=-ones(dim,1);
>> spy(A)
>> toll=1.e-10; nmax=1000;
>> u1=pcg(A,f,toll,nmax);
pcg converged at iteration 41 to a
  solution with relative residual 5.3e-11
```

Displacement $u(x)$



Now, try to reduce the number of iterations with a preconditioner.
A good preconditioner P:

- is a good image of A, for accelerating the convergence ($P^{-1}A \approx I$)
- is easy to solve, since it is solved at each iteration.

We start with the diagonal matrix extracted from A.

```
>> P=diag(diag(A));
>> u2=pcg(A,f,toll,nmax,P);
pcg converged at iteration 40
 to a solution with relative residual 9.9e-11
```

Low cost preconditioner $\Rightarrow$ **Little improvements**
Let us consider another preconditioner.
If we neglect bending forces (i.e. we consider only the matrix T), we are *actually modeling a vibrating string*. It is a "simplified model" however yielding a tridiagonal preconditioner.
In other words, we consider a simplified model as **preconditioner** for the complete one

This makes sense, since the preconditioner T is *tridiagonal* and *effective algorithms are available for solving triadiagonal matrices (e.g. Thomas method).*

```
>> P=1/h^2*T;
>> u3=pcg(A,f,toll,nmax,P);
pcg converged at iteration 24 to
 a solution with relative residual 2.1e-11
```

The effectiveness of the preconditioner is evident and can be explained by checking the condition number $\mathcal{K}(A)$ of the matrix.

```
>> format long e
>> cond(A)
ans =  1.795709678066970e+05

>> P=diag(diag(A));
>> cond(inv(P)*A)
ans =  1.795709678064624e+05

>> P=1/h^2*T;
>> cond(inv(P)*A)
ans = 5.406024737538536e+02
```


An advection-diffusion problem

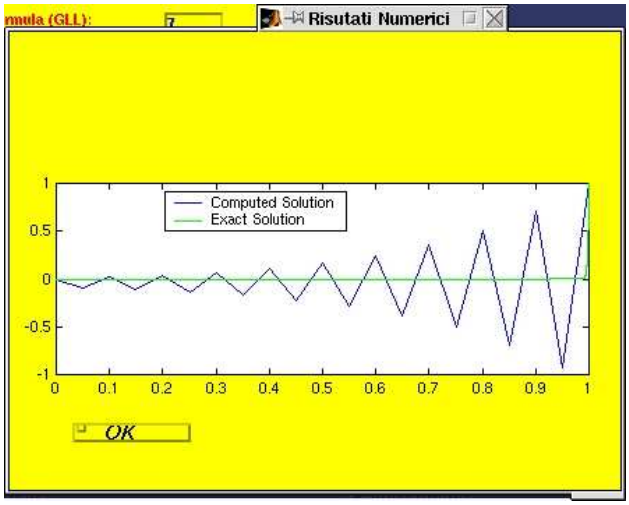
Consider the following problem:

$$\begin{aligned} -u'' + 500u' &= 0 \quad x \in (0,1) \\ u(0) = 0; &\quad u(1) = 1. \end{aligned} \tag{3}$$

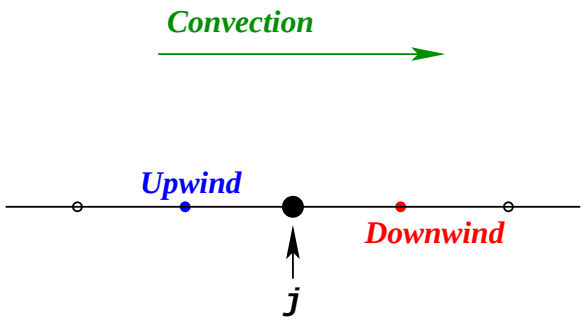
The red term represents the diffusion, the blue one the convection.
The exact solution of this problem is:

$$u(x) = \frac{e^{500x} - 1}{e^{500} - 1}. \tag{4}$$

If we try to solve this problem with a **pure Galerkin Finite Element approach**, e.g. with $h = 0.05$, we obtain an unsatisfactory solution.



In this problem, the convective term is clearly more relevant than the diffusive one (500 vs 1).
The convection is not symmetric, it **has a precise direction** ($500 > 0 \rightarrow$ **from left to right**).
However, the FEM scheme does not see the **direction**: the first order derivative is discretized giving in each node the same weight to Upwind and Downwind nodes.



This is the reason of these oscillations, that can be reduced or even eliminated by refining the mesh.
More precisely, if μ denotes the diffusion, $\mathbf{b}$ the convective field and the mesh size h is such that $h < \dfrac{2\mu}{\|\mathbf{b}\|}$ or equivalently the Péclet number is such that $\mathrm{Pe} := \dfrac{\|\mathbf{b}\|h}{2\mu} < 1$ the numerical solution does not oscillate.
In the case of the previous simulation: $\mathrm{Pe} \equiv \dfrac{bh}{2\mu} = \dfrac{500 \times 0.05}{2 \times 1} = 12.5 > 1.$
For the convection-diffusion of the oxygen in blood, $\dfrac{2\mu}{\|\mathbf{b}\|}$ is of order 10^{-6}, so a million of nodes should be needed for a simulation over a unit interval !!!
$\Rightarrow$ **Modifications of the pure Galerkin FE approach are in order** for eliminating oscillations without resorting to very small mesh sizes.

The basic idea is to introduce a numerical viscosity which:

- vanishes when $h \to 0$ so that the limit of the numerical problem still remains the continuous original mathematical problem (**consistency**)
- increases the real Peclet number, so that oscillations are eliminated (**stabilization**).

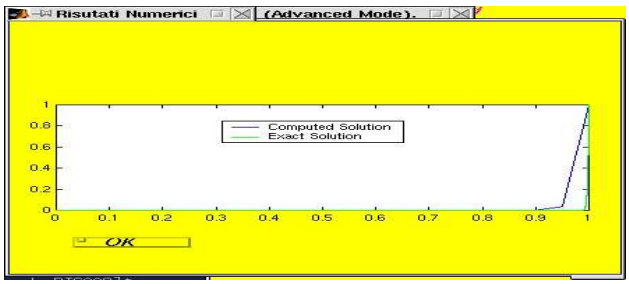
The numerical perturbation will introduce an **over-diffusion**, i.e. a *loss of accuracy*. This is the price to pay to the numerical stability.

Different methods have been proposed, for the introduction of numerical viscosity. We remind in 1D:

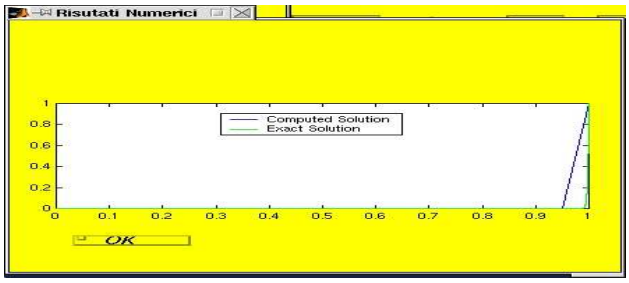
- `Upwind`
- `Scharfetter-Gummel`

The second approach is *less perturbative*, yielding stabilization with a smaller numerical viscosity than the Upwind one.

Upwind



Scharfetter-Gummel



An elliptic problem in 2D

Consider the following problem:

$$-\Delta u = \pi^2 \sin(\pi y) - 6(x-1)H(x-1) \tag{5}$$

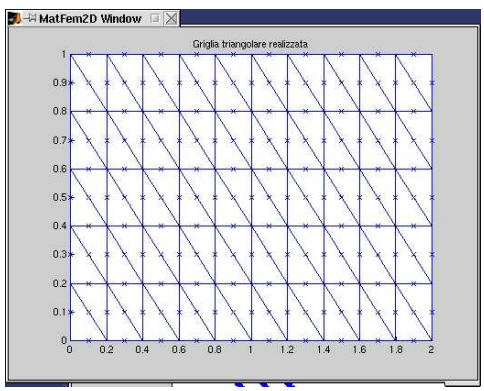
where $H(x)$ is the Heaviside step function, such that:

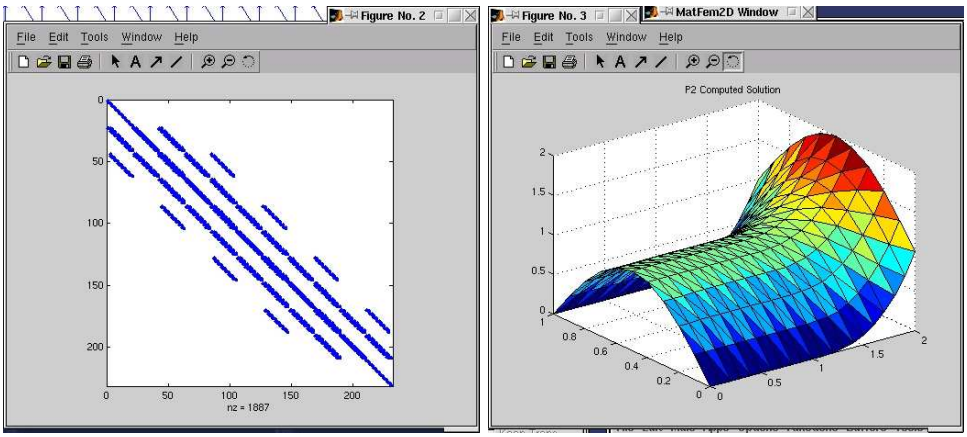
$$H(x) \begin{cases} 0 & \text{for } x < 0; \\ \\ 1 & \text{for } x > 0. \end{cases}$$

The computational domain is the $\Omega = (0,2) \times (0,1)$ with Dirichlet boundary conditions:

$$u = \sin(\pi y) + (x-1)^3 H(x-1). \tag{6}$$

It is possible to prove that $u(x,y) \in H^3(\Omega)$.





Error Analysis:

	$h = 0.2$	$h = 0.1$	$h = 0.05$
P1	0.600956	0.301583	0.150929
P2	0.047083	0.0117509	0.00294124

Consider now the following modification:

$$-\Delta u = \pi^2 \sin(\pi y) - 2H(x-1) \tag{7}$$

in the domain $\Omega = (0,2) \times (0,1)$ with the exact solution:

$$u = \sin(\pi y) + (x-1)^2 H(x-1). \tag{8}$$

It is possible to prove that $u(x,y) \in H^2(\Omega)$.

Error Analysis:

	$h = 0.2$	$h = 0.1$	$h = 0.05$
P1	0.57864	0.290369	0.145325
P2	0.0465805	0.0121859	0.00349274

An advection dominated problem in 2D

Consider the 2D problem:

$$\begin{aligned} -\Delta u + \mathbf{b} \cdot \nabla u &= f \qquad (x,y) \in (0,1)^2 \\ u &= 0 \qquad \text{on } \partial\Omega \end{aligned} \tag{9}$$

with $\mathbf{b} = [5000, 5000]^T$ and $f = 0$.
This is still an **advection-dominated** problem.
Also in 2D and 3D **pure Galerkin solution** are affected by oscillations if Pe> 1.
Also in these cases, the "**numerical therapy**" is the introduction of a suitable numerical viscosity.
Care must be taken for introducing viscosity only where needed, i.e. in the **wind direction**, not in the **Crosswind one**.

In particular, there is a class of methods (**strongly consistent methods**) which stabilize the numerical solution with the introduction of perturbations **vanishing when the exact solution is forced into the numerical problem.**
This is possible by modifying the **weak formulation** of the problem:

$$a(u_h, v_h) = \mathcal{F}(v_h)$$

becomes (Generalized Galerkin problem):

$$a(u_h, v_h) + \mathcal{R}(u_h, f, v_h) = \mathcal{F}(v_h)$$

where

$$\mathcal{R}(u_h, f, v_h) = \sum_{k \in \text{Elements of the mesh}} \delta_k \left(f - (-\Delta u_h + \mathbf{b} \cdot \nabla u_h), \Phi(v_h) \right).$$

BASIC FACTS:

- *Element by element the perturbing term depends on the residual of the (strong) problem*: $\Rightarrow$ the perturbation vanishes if $u_h = u_{exact}$.
- δ_K is a suitable scaling parameter.
- Different choices for $\Phi(v_h)$ give different *strongly consistent* methods. For instance:

  $$\text{SUPG} : \Phi(v_h) = \mathbf{b} \cdot \nabla v_h + \frac{1}{2} (\nabla \cdot \mathbf{b}) v_h$$

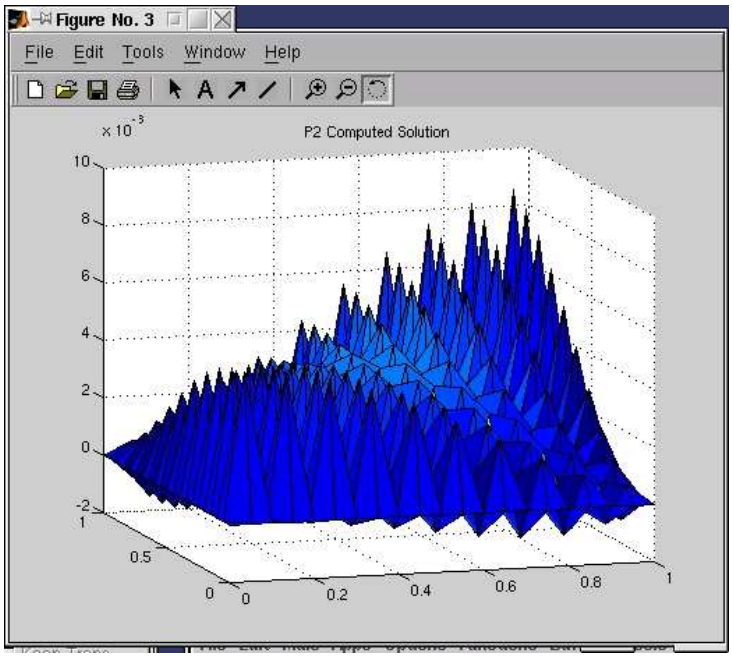
  In the present problem ($\nabla \cdot \mathbf{b} = 0$): $\boldsymbol{\Phi(\mathbf{v_h}) = \mathbf{b} \cdot \nabla \mathbf{v_h}}$

  $$\text{GALS} : \Phi(v_h) = -\Delta v_h + \mathbf{b} \cdot \nabla v_h$$

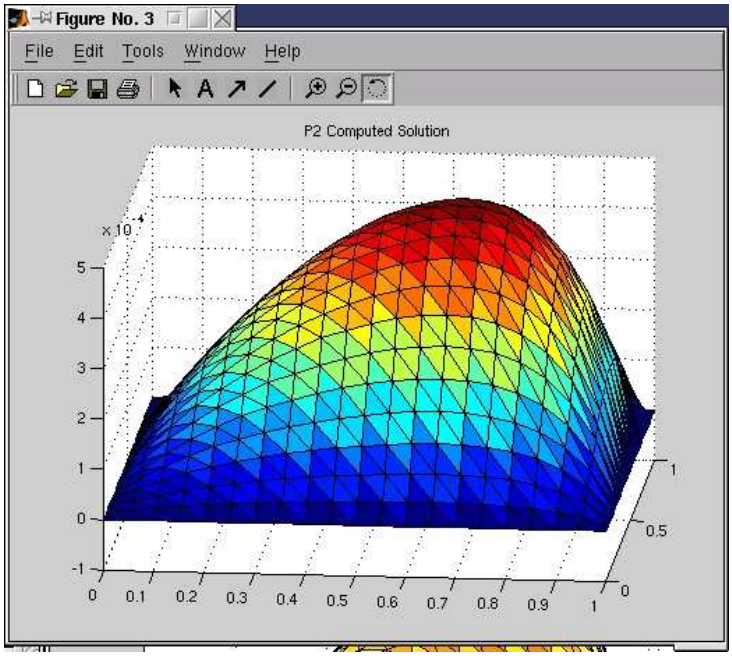
- The previous expressions, involving the strong formulation of the problem, make sense since the differential operators apply (element by element) to polynomial functions.

$\Rightarrow$ These methods are by far more accurate than Upwind.

`Galerkin` solution of a 2D advection-dominated problem.



`Upwind` solution of the 2D advection-dominated problem.



SUPG solution.