Numerical Methods

PhD Program in Natural Risks Assessment and Management (NRAM)

Spring 2012

Paola Gervasio

e-mail: gervasio@ing.unibs.it web: http://dm.ing.unibs.it/gervasio
tel.: 030 371 5734

What is Numerical Analysis

Numerical Analysis is a part of mathematics that investigates and studies efficient methods for computing approximated solutions of physical problems modeled by mathematical systems.

- Mathematical language
- Computers
- Approximation Techniques
- Efficiency

Some physical problems:

Fluid dynamics
Goal: Weather forecast, tsunami prevention
Goal: monitoring and reporting the spread of ashes
Goal: simulation of the fluid dynamics around obstacles

Offshore Platform Design
Goal: to develop a sound, reliable, and safe operating platform

Ice Sheet modeling
Goal: to model the three-dimensional motion of a glacier

...

Google search
Goal: quick and careful surfing on the web

Image compression and refinement
Computer graphics, rendering, textiles simulation,...

Differential operators

Divergence

$$\nabla \cdot \mathbf{v} = \frac{\partial v_1}{\partial x_1} + \frac{\partial v_2}{\partial x_2} + \frac{\partial v_3}{\partial x_3}$$

Gradient

$$\nabla v = \left[\frac{\partial v}{\partial x_1}, \frac{\partial v}{\partial x_2}, \frac{\partial v}{\partial x_3} \right]^t$$

Laplace operator

$$\Delta v = \frac{\partial^2 v}{\partial x_1^2} + \frac{\partial^2 v}{\partial x_2^2} + \frac{\partial^2 v}{\partial x_3^2}.$$

From the problem ($\longrightarrow$ mathematics $\longrightarrow$) numerical solution

1. Physical problem: flow in a channel

Data: geometry, inflow velocity, viscosity of the fluid, external forces (if any)

From the problem ($\rightarrow$ mathematics $\rightarrow$) numerical solution

1. Physical problem: flow in a channel

2. Mathematical model: Incompressible Navier-Stokes equations
   we want to look for velocity $\mathbf{v} = \mathbf{v}(t, \mathbf{x})$ and pressure $p = p(t, \mathbf{x})$, which are the solutions of

$$\begin{cases} \dfrac{\partial \mathbf{v}}{\partial t} - \nu \Delta \mathbf{v} + \mathbf{v} \cdot (\nabla \mathbf{v}) + \nabla p = \mathbf{f} & \text{in } \Omega \times (0, T) \\ \nabla \cdot \mathbf{v} = 0 & \text{in } \Omega \times (0, T) \\ \text{b.c. } + \text{ i.c.} \end{cases}$$

From the problem ($\longrightarrow$ mathematics $\longrightarrow$) numerical solution

1. Physical problem: flow in a channel

2. Mathematical model: Incompressible Navier-Stokes equations

3. Numerical approximation and algorithms
   approximation of derivatives, integrals, ...
   a small piece of a matlab program:

```
 for t = tspan(1)+dt:dt:tspan(2)
fn1 = f(xh,t);
rhs = An*[un;vn]+[dt*dt*zeta*fn1+dt*dt*(0.5-zeta)*fn;
dt*theta1*fn1+theta*dt*fn1];
rhs(1) = g(xspan(1),t); rhs(N)=g(xspan(2),t);
rhs(N+1:end)=rhs(N+1:end)-An211*temp1-An212*temp2;
uh = L\rhs; uh = U\uh;
fn = fn1; un = uh(1:N); vn = uh(N+1:end); uhg=[uhg,un];
end
```

From the problem ($\longrightarrow$ mathematics $\longrightarrow$) numerical solution

1. Physical problem: flow in a channel

2. Mathematical model: Incompressible Navier-Stokes equations

3. Numerical approximation and algorithms
   The discrete solution is $(\mathbf{v}_h, p_h)$, it depends on the discretization size h and in general it does not coincide with the exact solution $(\mathbf{v}, p)$.

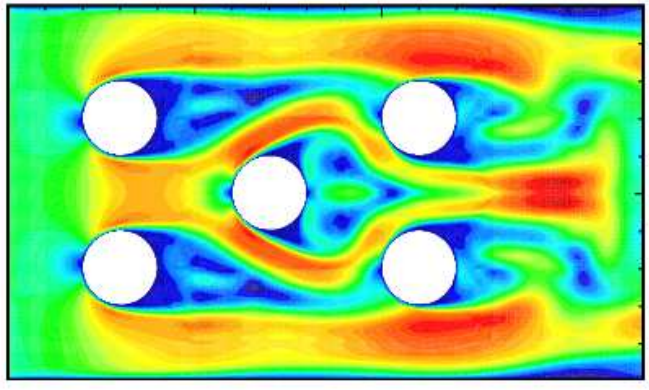
From the problem ($\longrightarrow$ mathematics $\longrightarrow$) numerical solution

1. Physical problem: flow in a channel

2. Mathematical model: Incompressible Navier-Stokes equations

3. Numerical approximation and algorithms
   The discrete solution is $(\mathbf{v}_h, p_h)$, it depends on the discretization size h and in general it does not coincide with the exact solution $(\mathbf{v}, p)$.

4. Postprocessing analysis:



The continuous problem

Mathematical Analysis

Is the problem well posed?

That is: there exists a unique solution and does it continuously depend on data? (= Do small changes on data provide small changes on the solution?)

How can we find the solution?

Mathematical analysis often does not provide explicit rules to find the solution, or sometimes the rules exist but are very complex to be implemented

We have to look for approximated solutions by computers

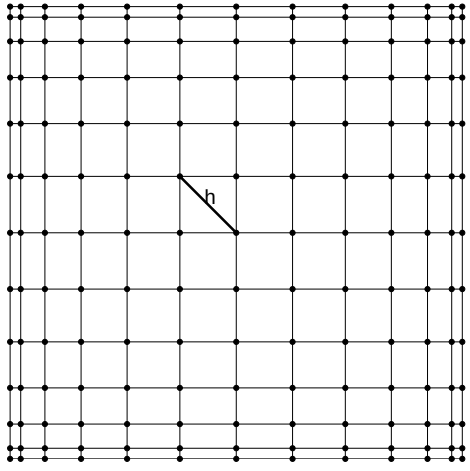
But digital computers are not able to compute exact derivatives, integrals and all of what refers to the "limit" $\lim_{t \to 0} f(t)$

Attention: We are not interested in computing derivatives and integrals by symbolic tools (e.g.. Derive, Maple, Mathematica, ...) since they do not go beyond the rules of mathematical analysis.

From mathematical analysis to numerical analysis

We have to translate the continuous problem in a discrete one:

1. Domain discretization: choose some points in the domain (a few: $\simeq 10^2$, lots of: $\simeq 10^6$, but the number is always finite) at which to look for the approximated (or numerical) solution. h is the discretization "size"



From mathematical analysis to numerical analysis

We have to translate the continuous problem in a discrete one:

1. Domain discretization
2. Derivative approximation by finite differences (or any other method: finite elements, spectral elements, finite volumes, etc.):
   $\frac{\partial f}{\partial x}(x_0) \simeq \frac{f(x_0+h)-f(x_0)}{h}$ where h is small
   $\left(\frac{\partial f}{\partial x}(x_0) := \lim_{h\to 0} \frac{f(x_0+h)-f(x_0)}{h}\right)$
3. Solve linear (or non linear) systems of equations

$$A \begin{bmatrix} \mathbf{v}_h \\ p_h \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0 \end{bmatrix}$$

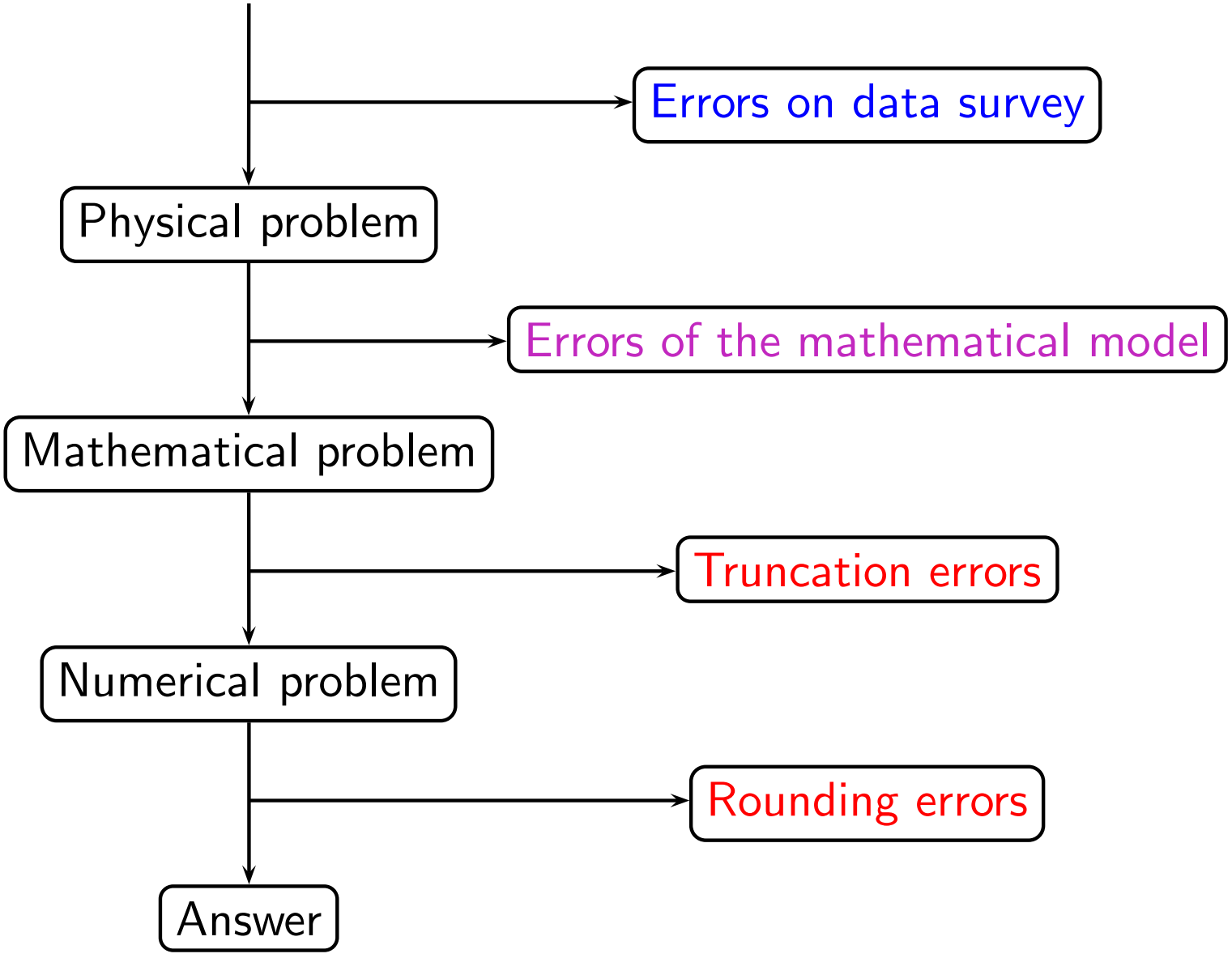
4. Rebuild the solution in the remainder of the computational domain starting from $\mathbf{v}_h$, and p_h.

Which are the goals of Numerical Analysis?

1. To propose discretization methods for mathematical problems.
2. To investigate if, when $h \longrightarrow 0$, the numerical solution $(\mathbf{v}_h,\ p_h)$ converges to the exact one $(\mathbf{v}, p)$ (theoretical study)
3. To translate numerical methods in efficient (in terms of CPU-time and memory allocation) and stable (less sensible as possible to machine arithmetic) algorithms
4. To verify the results

Ween need both mathematical and computational knowledges

Several types of errors



Truncation (or approximation) errors

In approximating $f'(x_0) \simeq \dfrac{f(x_0+h)-f(x_0)}{h}$ we introduce an error.

Write the Taylor 1st-order expansion of $f(x)$ centered in x_0:

$$f(x) = f(x_0) + f'(x_0)(x-x_0) + \frac{f''(\xi)}{2}(x-x_0)^2.$$

Set $x = x_0 + h$:

$$f(x_0+h) = f(x_0) + f'(x_0)(x_0+h-x_0) + \frac{f''(\xi)}{2}(x_0+h-x_0)^2,$$

explicit $f'(x_0)$ and it holds:

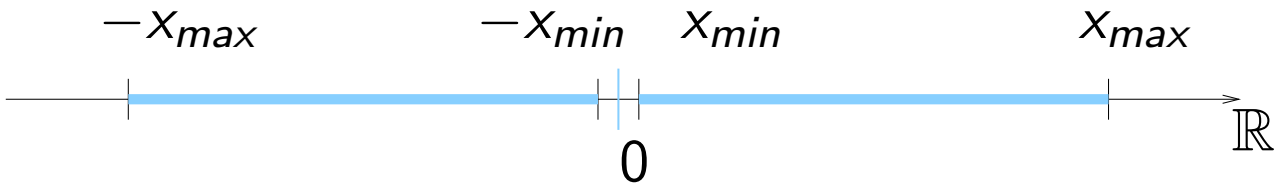
$$f'(x_0) = \frac{f(x_0+h)-f(x_0)}{h} - \frac{f''(\xi)}{2}h$$

The error introduced in this approximation is of truncation type since we have cut off the Taylor expansion of f to approximate f'.

Machine arithmetic

A computer cannot represent any real number, but only those with a finite number of digits lying in the bounded set

$$[-x_{max}, -x_{min}] \cup \{0\} \cup [x_{min}, x_{max}]$$



In Matlab: $x_{min} = 2.2251 \cdot 10^{-308}$, $x_{max} = 1.7977 \cdot 10^{308}$.
Matlab commands to print these numbers are `realmin` and `realmax`
Matlab uses the basis $\beta = 2$ and 8 Bytes to store a floating point number.

Rounding errors

To represent a number, the computer introduces an error (rounding error) characterized by the basis β and the number t of bits used to store the mantissa of the number itself:

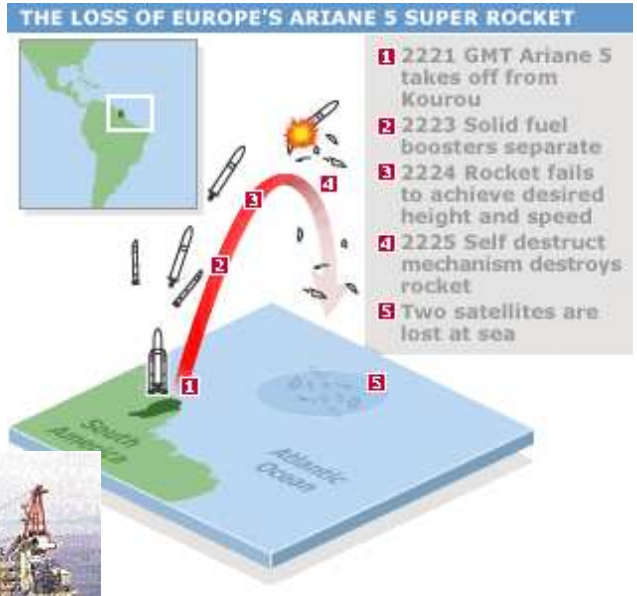
$$|x - fl_t(x)| \leq \frac{1}{2}\beta^{1-t}|x|$$

Matlab uses $\beta = 2$ and $t = 53$,

$$\frac{1}{2}\beta^{1-t} = 1.1102 \cdot 10^{-16}$$

Non-trivial errors

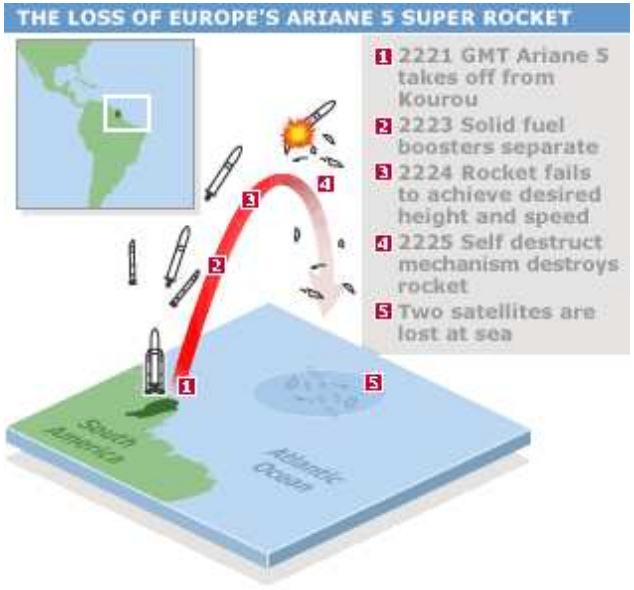
What do they share?



February, 25 1991: A Patriot missile fails in hitting a SCUD missile, the latter falls on U.S. barrack: 28 people died.

Why?

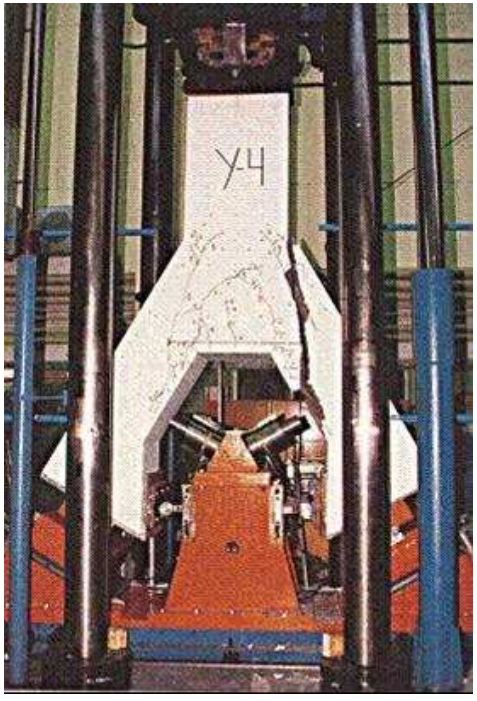
The excessive propagation of rounding errors (in measuring time) gives rise to a large error in computing the trajectory of the missile.



June 1996: The Ariane 5 missile explods 37 seconds after it started

Why?

A false overflow warning causes the selfcontrol system has gone haywire



The offshore platform folds down in the testing phase

Why?

Inaccurate simulation: truncation errors too large

The translation from the continuous problem to the discrete problem is sometimes difficult to do

It is not guaranteed that a good theoretical method is good and efficient in practice

A recursive inefficient formula

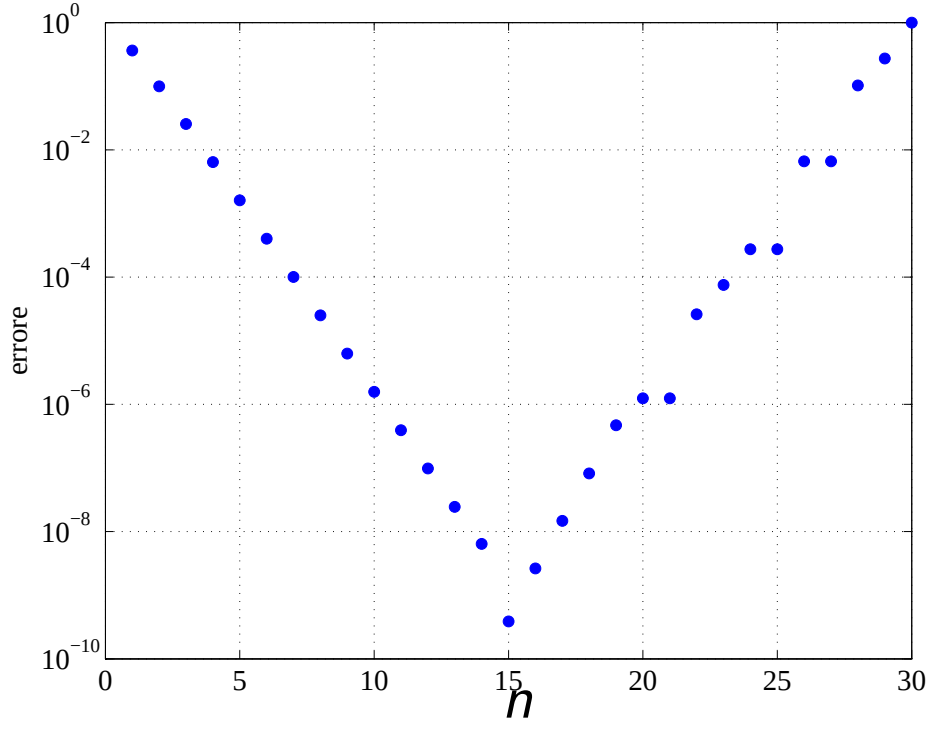
$$\begin{cases} f_2 = 2 \\ f_{n+1} = 2^{n-0.5}\sqrt{1-\sqrt{1-4^{1-n}f_n^2}}, \quad n = 2, 3, ... \end{cases}$$

It holds that $\lim_{n\to\infty} f_n = \pi$

Write a matlab function to implement the recursive formula:

```
f(1)=2; err(1)=abs(pi-2)/pi;
for n=2:nmax
f(n)=2^(n-0.5)*sqrt(1-sqrt(1-4^(1-n)*f(n-1)^2));
err(n)=abs(pi-f(n))/pi;
if err(n) <= 1.e-15
break
end
end
```

Numerical results



The theoretical formula DOES NOT work well in practice because of the propagation of rounding errors

Another type of inefficiency

The goal is to solve a linear system of 25 equations and 25 unknowns by the Cramer rule on the fastest computer in the world (at november 2011, it was Fujitsu K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect with 705024 Cores, 10.5 Petaflops $= 10.5 \times 10^{15}$ floating-point operations per second)

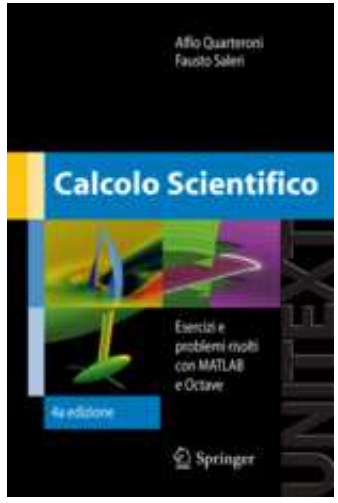
We must do about $3 \cdot 26! \simeq 4.03 \cdot 10^{26}$ floating point operations, that is

38365 years

Note that the the performance of an Intel Core i7-975 Desktop processor (2.66 GHz) is about 42 GFlops ($42 \cdot 10^9$ flops).
We need numerical methods that are faster and more efficient than classical ones. Luckily, by using the most efficient numerical methods known today, a linear system 25×25 can be solved in 10^{-4} seconds by our desktop!

References

A. Quarteroni, F. Saleri.
Calcolo Scientifico,
4a ed. Springer Italia, Milano, 2008.



A. Quarteroni, F. Saleri, P. Gervasio.
Scientific Computing with Matlab and Octave,
3rd ed. Springer, Berlin, 2010.