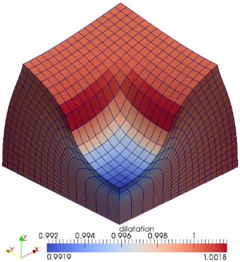
Specialized Numerical Methods for Transport Phenomena

The finite element method: Non-linear Problems

Bruno Blais

Associate Professor
Department of Chemical Engineering
Polytechnique Montréal

October 22, 2025



Outline

Outline

Non-linear problems in engineering

Formalism

A straigthforward approach: Picard's method

Pseudo-transient method

Newton's method: a review

- Newton-Raphson Method

Newton's method in FEM

Conclusion

Motivation

So far, we have learnt how to solve a linear PDE, the Poisson equation. It is a common PDE that enables us to model many phenomena:

- Heat diffusion
- Diffusion of chemical species
- Diffusion of electrons
- Linear elasticity
- etc...

However, the majority of the problems of interest in engineering are not linear. Solving these problems will be significantly more involved.

Temperature dependent properties

In reality, the thermal conductivity of a material often depends on its temperature. If we assume that the thermal conductivity k varies linearly with temperature:

$$k = A + BT$$

Then our heat equation will be more complex

$$\nabla \cdot (k \nabla T) = 0$$

Let's break this down...

Chemical reactions

Many chemical reactions have non-linear kinetics. For example, the following reaction:

$$2\,\mathrm{A} \longrightarrow \mathrm{B}$$

would lead to the following diffusion-reaction equation:

$$\nabla \cdot (D \nabla c_{\mathsf{A}}) = -k c_{\mathsf{A}}^2$$

What makes this equation non-linear?

Euler equations

$$
\begin{aligned}
\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_x}{\partial x} + \frac{\partial \rho u_y}{\partial y} &= 0 \\
\frac{\partial \rho u_x}{\partial t} + \frac{\partial \rho u_x u_x}{\partial x} + \frac{\partial \rho u_y u_x}{\partial y} + \frac{\partial p}{\partial x} &= 0 \\
\frac{\partial \rho u_y}{\partial t} + \frac{\partial \rho u_x u_y}{\partial x} + \frac{\partial \rho u_y u_y}{\partial y} + \frac{\partial p}{\partial y} &= 0 \\
\frac{\partial \rho e}{\partial t} + \frac{\partial \left(\rho e + \frac{1}{2}\rho \left|\boldsymbol{u}\right|^2 + p\right) u_x}{\partial x} \frac{\partial \left(\rho e + \frac{1}{2}\rho \left|\boldsymbol{u}\right|^2 + p\right) u_y}{\partial y} &= 0
\end{aligned}
$$

Navier-Stokes equations

$$\nabla \cdot \boldsymbol{u} = 0$$

$$\rho \frac{\partial \boldsymbol{u}}{\partial t} + \rho \left(\boldsymbol{u} \cdot \nabla \right) \boldsymbol{u} = -\nabla p + \mu \nabla^2 \boldsymbol{u} + \rho \boldsymbol{g}$$

Many others...

- Cahn-Hilliard
- Saint-Venant
- Plasticity
- ...

Outline

Notation

We will write all non-linear problems in the same form, which we will call the residual form:

$$\boldsymbol{R}(\boldsymbol{v}) = 0$$

where $\boldsymbol{v}$ is the vector of state variables of the problem. For example, for the temperature equation: $\boldsymbol{v} = [T]$. $\boldsymbol{R}(\boldsymbol{v})$ is the residual.

Our non-linear equation will have been solved when we have found $\boldsymbol{v}^*$ such that $\boldsymbol{R}(\boldsymbol{v}^*) = 0$.

Example of notation

Take the following equation:

$$\nabla^2 T = cT$$

with c a constant. This equation written in residual form is:

$$\nabla^2 T - cT = \boldsymbol{R}(T)$$

when $\boldsymbol{R}(T) = 0$, we have solved our equation.
Is this equation linear or non-linear?

Outline

Picard's method

Picard's method (or successive approximation) is a method to solve non-linear equations that is very easy to implement. Let's illustrate how it works using an example.

Example

Recall our temperature dependent thermal conductivity problem.

$$k = A + BT$$

with our heat transfer equation

$$\nabla \cdot (k \nabla T) = 0$$

The issue with this equation is that the conductivity depends on T leading to a non-linearity.

Example

Recall our temperature dependent thermal conductivity problem.

$$k = A + BT$$

with our heat transfer equation

$$\nabla \cdot (k \nabla T) = 0$$

The issue with this equation is that the conductivity depends on T leading to a non-linearity.

The idea behind Picard's method is to solve it successively by estimating the weaker non-linear terms at a previous value of the solution.

Successive approximations

In this context, Picard's method would take the following form:

Input:

- Initial estimate of the temperature T^0
- Tolerance ϵ and maximal number of iterations N

Result: The field T which satisfies the non-linear equation.

Initialize $n = 0$ and $T^0 = 1$

while $|\boldsymbol{R}| > \epsilon$ ***and*** $n < N$ **do**

  Calculate T^{n+1} by solving:

  $\nabla \cdot \left(k(T^n)\nabla T^{n+1}\right) = 0$

  Calculate $\boldsymbol{R}(T^{n+1})$

  $n = n + 1$

end

Conclusions

Advantages

- Very easy to implement
- Converges in general for weakly non-linear problems
- Can be coupled to relaxation methods to ensure stability

Disadvantages

- The choice of initial estimate is very important
- Convergence can be **very slow**. More than 10 iterations may be required. If relaxation is used, this is even worse

Outline

Concept

Pseudo transient (or adjoint time-stepping) methods offer a slow, but robust way of solving non-linear problems.

- Formulate the non-linear problem as a transient problem
- Make a component of the non-linearity explicit
- Integrate in time until steady-state is reached

Example

Let's take our example problem:

$$\nabla \cdot (k \nabla T) = 0$$

We transform it to its transient counterpart:

$$\frac{\partial T}{\partial t} - \nabla \cdot (k(T) \nabla T) = 0$$

we apply our finite difference scheme in time and formulate an explicit expression for k.

$$\frac{T^{t+\Delta t} - T^{t}}{\Delta t} - \nabla \cdot \left(k(T^{t}) \nabla T^{t+\Delta t} \right) = 0$$

Analysis

Advantages

- Robust. The lower the time step, the more robust this is. It behaves similar to a relaxed successive approximation
- Not too difficult to implement

Disadvantages

- Slow. May require prohibitively small time steps

Outline

Concept

We need a faster method to solve non-linear problems. We will leverage a well-established method which you might have seen in the past: Newton's method.

- Let's recap how it works for regular problems
- Then let's explain how it works in the context of the finite element method

Towards Newton's method

Taylor series

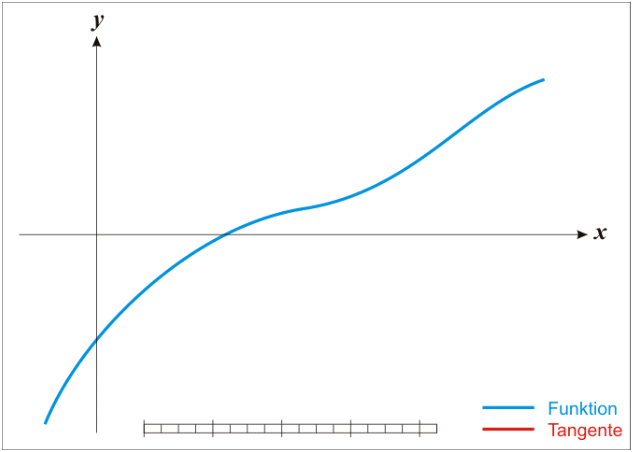
The Taylor series of a real function $f(x)$ around a point x where the function is differentiable is:

$$f(x+\delta x) = f(x) + \frac{\delta x}{1!} f'(x) + \frac{(\delta x)^2}{2!} f''(x) + \frac{(\delta x)^3}{3!} f'''(x) + \ldots$$

We seek to construct a systematic method for solving equations of the form:

$$f(x) = 0$$

Graphical interpretation



The derivative of the function is used to draw a line to approximate the root. The closer the point of evaluation of the derivative is to the root, the better the approximation.

Algorithm

Input:

- An initial estimate x^0
- A function that calculates $f(x)$ and another which calculates $f'(x)$
- A tolerance ϵ and a maximal number of iterations N

Result: The root r such that $f(r) \approx 0$

Initialize $n = 0$ and $\delta x^0 = 1$

while $|\delta x^n| > \epsilon$ ***and*** $n < N$ **do**

    Calculate $\delta x^n = -\frac{f(x^n)}{f'(x^n)}$

    Calculate $x^{n+1} = x^n + \delta x^n$

    $n = n + 1$

end

Newton-Raphson

The Newton-Raphson method is the generalization of Newton's method for d equations. Consider the following system of equations:

$$f_1(x_1, x_2) = 0$$
$$f_2(x_1, x_2) = 0$$

Remark: we will note here the coordinate with the subscript and the iteration with the exponent. So x_k^n is the coordinate $k \in [1, d]$ of the vector $\boldsymbol{x}$ at iteration n.

Taylor expansion for 2 equations

We can solve the system in the neighborhood of a point $\boldsymbol{x}_i$.

$$
\begin{aligned}
f_1(x_1 + \delta x_1, x_2 + \delta x_2) = 0 \\
f_2(x_1 + \delta x_1, x_2 + \delta x_2) = 0
\end{aligned}
$$

The Taylor expansion of these two functions is the natural extension of the 1D case.

$$
\begin{aligned}
f_1(x_1 + \delta x_1, x_2 + \delta x_2) = f_1(x_1, x_2) + \frac{\partial f_1(x_1, x_2)}{\partial x_1} \delta x_1 + \frac{\partial f_1(x_1, x_2)}{\partial x_2} \delta x_2 + ... \\
f_2(x_1 + \delta x_1, x_2 + \delta x_2) = f_2(x_1, x_2) + \frac{\partial f_2(x_1, x_2)}{\partial x_1} \delta x_1 + \frac{\partial f_2(x_1, x_2)}{\partial x_2} \delta x_2 + ...
\end{aligned}
$$

Taylor expansion for 2 equations

Neglecting the high order terms, we obtain the linear approximation of the system of nonlinear equations on which we can iterate:

$$
\begin{aligned}
f_1(x_1, x_2) + \frac{\partial f_1(x_1, x_2)}{\partial x_1}\delta x_1 + \frac{\partial f_1(x_1, x_2)}{\partial x_2}\delta x_2 = 0 \\
f_2(x_1, x_2) + \frac{\partial f_2(x_1, x_2)}{\partial x_1}\delta x_1 + \frac{\partial f_2(x_1, x_2)}{\partial x_2}\delta x_2 = 0
\end{aligned}
$$

Introducing the residual vector $\boldsymbol{R}(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x})$, the jacobian matrix $\boldsymbol{\mathcal{J}}(\boldsymbol{x})$ and the correction vector $\boldsymbol{\delta x}$ we obtain:

$$
\mathcal{J}(\boldsymbol{x})\boldsymbol{\delta x} = -\boldsymbol{R}(\boldsymbol{x})
$$

As in the case of the classical Newton method, the Newton-Rapshon method will converge when the norm of the residual is below a fixed tolerance.

Algorithm

Input:

- A starting vector $\boldsymbol{x}^0$
- A function to calculate $\boldsymbol{R}(\boldsymbol{x})$ and $\mathcal{J}(\boldsymbol{x})$
- A tolerance ϵ and a maximal number of iterations N

Result: The root $\boldsymbol{r}$ such that $\|\boldsymbol{R}(\boldsymbol{r})\| \approx 0$

Initialize $n = 0$ and $\boldsymbol{\delta x}^0 = 1$

while $\|\boldsymbol{\delta x^n}\| > \epsilon$ ***and*** $n < N$ **do**

  Calculate $\mathcal{J}(\boldsymbol{x}^n)$ and $\boldsymbol{R}(\boldsymbol{x}^n)$

  Solve the linear system $\mathcal{J}(\boldsymbol{x}^n)\boldsymbol{\delta x}^n = -\boldsymbol{R}(\boldsymbol{x}^n)$

  Calculate $\boldsymbol{x}^{n+1} = \boldsymbol{x}^n + \boldsymbol{\delta x}^n$

  $n = n + 1$

end

Let us talk about relaxation methods...

Outline

Using Newton's method in FEM

Newton's method is an effective way to solve non-linear equations. It requires the following:

- Expression of the residual
- Expression of the jacobian matrix

Calculating the Jacobian matrix in the context of the finite element method is not as easy as it seems.

Let's see with an example!

Example

We wish to solve the following problem:

$$\nabla \cdot (k\nabla T) - T^2 = 0$$

There are two ways we can proceed:

1. Obtain the weak form of the non-linear problem, then calculate the Jacobian matrix
2. Calculate the Jacobian of the non-linear problem by linearizing it, then obtain the weak form

Both approaches are strictly equivalent, but it is easier not to get confused with the second one. Let's proceed.

Calculating the derivative

We want to solve:

$$\nabla \cdot (k \nabla T) - T^2 = 0$$

Using Newton's method:

$$\mathcal{J}(\boldsymbol{x}) \boldsymbol{\delta x} = -\boldsymbol{R}(\boldsymbol{x})$$

How do I calculate $\mathcal{J}(\boldsymbol{x})$? Using a directional derivative:

$$\mathcal{J}(\boldsymbol{x}) \boldsymbol{\delta x} = \lim_{\epsilon \to 0} \frac{\boldsymbol{R}(\boldsymbol{x} + \epsilon \delta \boldsymbol{x}) - \boldsymbol{R}(\boldsymbol{x})}{\epsilon}$$

Solution

The residual is:

$$R(T) = \nabla \cdot (k \nabla T) - T^2 = 0$$

The directional derivative is:

$$J(T)\delta T = \nabla (k \nabla \delta T) - 2T\delta T$$

Towards the weak form

The problem we now want to solve is:

$$\nabla \cdot (k \nabla \delta T) - 2T\delta T = -R(T)$$

What is our unknown now? It is not T anymore, but it is the correction δT.
This is the variable we will be solving for. It is also the variable we need boundary conditions for!

Solution

The weak form of the problem, assuming Dirichlet boundary conditions, is:

$$-\sum_j \delta T_j \left(\iiint\limits_{\Omega} k\nabla\phi_i\nabla\phi_j + \iiint\limits_{\Omega} 2T\phi_i\phi_j \right) = \left(\iiint\limits_{\Omega} k\nabla\phi_i\nabla T + \iiint\limits_{\Omega} \phi_i T^2 \right)$$

by solving this problem iteratively, we will reach a point where $R(T)$ will become zero and vanish.

Other examples

Try to calculate the directional derivatives of the following as exercise:

$$\begin{aligned} \nabla \cdot (T \nabla T) &= 0 \\ \nabla \cdot (\nabla T) &= \sin T \\ \nabla \cdot (\nabla T) &= \sigma (T^4 - T_\infty^4) \\ \nabla \cdot (\nabla c^3) &= 0 \end{aligned}$$

Outline

Conclusions

We have seen multiple methods to solve non-linear problems:

- Picard
- Pseudo-transient
- Newton-Raphson

All methods have their pros and cons. Although Newton's method is by far the most efficient, obtaining an analytical formulation of the Jacobian can be an immense challenge for some equations. It is always necessary to find a balance between human time and computational time!