Solving the Linear System Matrix: Direct and Iterative Solvers
Solving the Linear System Matrix

- Linear system matrix
- Direct methods
- Iterative methods
- Suggestions on solver selection
Choosing the linear system solver

- COMSOL chooses the optimized solver and its settings based on the chosen space dimension, physics and study type.

- We can also choose a different solver:
  - Not recommended in general
  - Could be useful for multiphysics problems
Let’s take a look at the system equations

\[
f(u) = \begin{cases} 
0 \\
p 
\end{cases} - \begin{bmatrix} k_2 + k_3 & -k_3 \\
-k_3 & k_1 + k_3 
\end{bmatrix} \begin{bmatrix} u_2 \\
u_3 
\end{bmatrix} = \begin{bmatrix} 0 \\
0 
\end{bmatrix} = b - Ku = 0
\]

Define a quadratic function: \( r(u) = b \cdot u - \frac{1}{2}Ku \cdot u \)

The solution, \( f(u_{\text{solution}}) = 0 \), is the point where \( r(u) \) is at a minimum.
Finding the minimum of a quadratic function

\[ r(u) = 2u^2 - 3u + 1 \]

Newton’s method:

\[
 u_{sol} = u_0 - \frac{r'(u_0)}{r''(u_0)}
\]

For a quadratic function, this converges in one iteration, for any \( u_0 \)

Via our choice of \( r(u) \), this reduces to the same equation from the previous section:

\[
 r(u) = b \cdot u^{-1/2} Ku \cdot u \\
 r'(u) = b - Ku \\
 u_0 = 0 \\
 u_{sol} = u_0 - \frac{r'(u_0)}{r''(u_0)} = u_0 - \frac{b - Ku_0}{K} = 0.75
\]

\[
 u_{sol} = 0 + \frac{b}{K} = K^{-1}b
\]
Finding the minimum of the quadratic function, \( r(u) \), by the direct method means solving \( u=K^{-1}b \)

- This is known as Gaussian Elimination, or LU factorization
- The numerical algorithms are beyond the scope of this course, but they have the following important properties:
  - For 3D, requires \( O(n^2) - O(n^3) \) numeric operations, where \( n \) is the length of \( u \)
  - Requires \( O(n^2) - O(n^3) \) memory
  - Robustness of the algorithm is only very weakly dependent upon \( K \)
- The direct solvers in COMSOL are:
  - MUMPS: fast, multi-core capable, cluster capable
  - PARDISO: fast, robust, multi-core capable (scales better than MUMPS on a single node with many cores)
  - SPOOLES: slow, uses the least memory, multi-core, cluster capable
An iterative method for finding the minimum: The Conjugate Gradient (CG) method

1) Start here
2) Initially, find the gradient vector
3) Find the minimum along that vector
4) Find the conjugate gradient vector
5) Repeat 3-4 until converged

The CG method requires that we can evaluate $r(u)$, $r'(u)$ and $r''(u)$

CG converges in at most $n$ iterations

CG does NOT compute $K^{-1}$

*) See e.g. Wikipedia, or “Scientific Computing” by Michael Heath
The numerical values in $K$ can affect the algorithm

$$r(u) = b \cdot u^{-\frac{1}{2}}Ku \cdot u$$

The higher the condition number the more numerical error creeps into the solution
All iterative methods in COMSOL are some variation upon the CG method

- Conjugate Gradient, GMRES, FGMRES, BiCGStab and Geometric Multigrid
  - The details of these algorithms are beyond the scope of this course

- All these methods make use of **preconditioners**

- The system equation, $Ku = b$, is multiplied by a preconditioner matrix, $M$, to improve the condition number

$Ku = b$  \quad  MKu = Mb$

Exercise: Show that the best possible preconditioner is the matrix $M=K^{-1}$
What you need to know about iterative solvers

- They converge in at most $n$ iterations \textit{(good)}
- Solution time is $O(n^{1-n^2})$ \textit{(good)}
  - Solution time depends on condition number
- Memory requirements are $O(n^{1-n^2})$ \textit{(very good)}
- They are less robust than the direct solvers \textit{(neutral)}
  - Convergence depends upon condition number
  - An ill-conditioned problem is often set up incorrectly
- Different physics require different iterative methods \textit{(bad)}
  - This is an ongoing research topic
  - Often cannot solve two physics with the same solver
  - Improvements in methods are ongoing
  - We have tried to find the best combination of iterative solvers and preconditioners for many physics, to find these settings, read the manuals or open a new model file, select a space dimension of 3D and the physics you want to solve
Example: Structural Mechanics

- 1MDOF Linear problem with good mesh quality
  - Direct: Use 16GB RAM and >1 hour to solve (2 cores)
  - GMRES+Geometric Multigrid+SOR
    - 3GB RAM, 166 seconds of solving time

- However: It needs a good mesh quality
- A bit hassle to setup, but with great advantage in speed, it might be worth it.
Definitions of various types of couplings

• **One-way coupled**
  – Information passes from one physics to the next, in one direction

• **Two-way coupled**
  – Information gets passed back and forth between physics

• **Load coupled**
  – The results from one physics affect only the loading on the other physics

• **Material coupled**
  – The results from one physics affect the materials properties of other physics

• **Non-linear coupled**
  – The results of one physics affects both that, and other, physics

• **Fully coupled**
  – All of the above

• **Weakly coupled**
  – The physics do not strongly affect the loads/properties in other physics

• **Strongly coupled**
  – The opposite of weakly coupled
Segregated solver

- Applicable to weakly coupled nonlinear problems
- Strongly nonlinear problems will suffer from slow or no convergence
- Use to solve iteratively between different solution variables.
- Example:
  - Fluid flow with transport of chemical species
  - Used in k-epsilon models for stabilizing the solution process
- Saves memory
- Can be very efficient
Using the segregated solver

Study > Solver Configurations > Solver 1 > Stationary Solver 1

Can add multiple Segregated steps as required.
Segregated step

Choose the variable/s that you want to solve in this segregated step.
Segregated step – more specifications

For each segregated step you can choose the linear system solver and damping and termination techniques.
Achieving convergence for multiphysics problems

• Set up the coupled problem and try solving it with a direct solver

• If it is not converging:
  – Check initial conditions
  – Ramp the loads up
  – Ramp up the non-linear effects
  – Make sure that the problem is well posed (this can be very difficult!)

• If solve time is long, or a lot of memory is needed
  – Use the segregated solver and select the optimal solver (direct or iterative) for each physics, or group of physics, in the problem
  – Upgrade hardware

• Perform a mesh refinement study
Recap: Working with solvers

- A problem could be Linear or Nonlinear

- The linear system of equations could be solved using a Direct Solver or an Iterative Solver

- A multiphysics problem could be solved using a Fully-Coupled Approach or Segregated Approach