ROYAL INSTITUTE OF TECHNOLOGY

ODEs, part 1

Anna-Karin Tornberg

Mathematical Models, Analysis and Simulation
Fall semester, 2010

A system in equilibrium

[Material from Strang, sections 2.1 and 2.2].
Consider a system of springs and masses in equilibrium.
[NOTES]

To obtain our system of equations, we have applied three equations:

1) The forces should be in equilibrium.
2) Hooke's law for springs.
3) Relation between spring length and mass positions x_i.

The system is on the form: $\mathbf{A} = \begin{bmatrix} -\mathbf{C}^{-1} & \mathbf{A} \\ \mathbf{A}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{F} \\ \mathbf{x} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ \mathbf{f} \end{bmatrix}$, where the columns of $\mathbf{A}$ are linearly independent, and $\mathbf{C}$ is a diagonal matrix with positive entries on the diagonal.

This yields $\mathbf{A}^T\mathbf{C}\mathbf{A}\mathbf{x} = \mathbf{f} + \mathbf{A}^T\mathbf{C}\mathbf{b}$.
"Stiffness matrix" $\mathbf{K} = \mathbf{A}^T\mathbf{C}\mathbf{A}$ is symmetric positive definite.
Strang works with displacements u_i instead of positions x_i. In this case, replacing $\mathbf{x}$ by $\mathbf{u}$ in the system, we will have $\mathbf{b} = 0$. [NOTES]

Minimization

We have

$$\mathbf{Ku} = \mathbf{f}, \text{ with } \mathbf{K} = \mathbf{A}^T\mathbf{CA}$$

From earlier:
The solution to $\mathbf{Ku} = \mathbf{f}$ is the $\mathbf{u}$ that minimizes the quadratic form $P(\mathbf{u}) = 1/2\mathbf{u}^T\mathbf{Ku} - \mathbf{u}^T\mathbf{f}$.

- Stretching increases the internal energy, $1/2\mathbf{e}^T\mathbf{Ce} = 1/2\mathbf{u}^T\mathbf{Ku}$. ($\mathbf{e}$ is the elongation of the spring).
- The masses loose potential energy by $\mathbf{f}^T\mathbf{u}$ (force times displacement, work done by external forces).

This is NOT a conserved system: external forces (in our example gravitational forces) are active.

Hence, the configuration of this system at equilibrium is the configuration that minimizes the potential energy of the system.

A dynamic system

Again, a line of springs. Now, not trying to find the equilibrium, but the evolution in time.

To obtain our system of equations, we will again apply three equations:

1) Newton's law.
2) Hooke's law for springs.
3) Relation between spring length and mass positions x_i.

2) and 3) the same as before. But instead of in 1) requiring that forces are in equilibrium, we use that a net force will give rise to an acceleration.

Compared to $\mathbf{Kx} = \mathbf{f} + \mathbf{A}^T\mathbf{Cb}$, or equivalently $\mathbf{Ku} = \mathbf{f}$ before, we now get [NOTES]:

$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{Kx} = \mathbf{f} + \mathbf{A}^T\mathbf{Cb}$, or equivalently $\mathbf{M}\ddot{\mathbf{u}} + \mathbf{Ku} = \mathbf{f}$, where $\ddot{\mathbf{x}}$ indicates two time derivatives of $\mathbf{x}$.
$\mathbf{M}$ is the "mass matrix" with the size of the masses on the diagonal.

Damping and non-dimensionalization

Let us consider the case with only one spring and one mass, where a dashpot yields a damping proportional to velocity. Let us further assume that we have an external oscillatory driving force.
We will non-dimensionalize the equations, and also discuss the cases of over damping, under damping and critical damping. [NOTES]

System of first order equations

A second order equation can be written as a system of first order equations.

$$\frac{d^2x}{dt^2} + c\frac{dx}{dt} + dx = f(t)$$

Let $\mathbf{u} = \begin{bmatrix} x \\ \frac{dx}{dt} \end{bmatrix}$. Then the equation can be written as

$$\frac{d}{dt}\underbrace{\begin{bmatrix} x \\ \frac{dx}{dt} \end{bmatrix}}_{\mathbf{u}} + \underbrace{\begin{bmatrix} 0 & -1 \\ d & c \end{bmatrix}}_{\mathbf{A}}\underbrace{\begin{bmatrix} x \\ \frac{dx}{dt} \end{bmatrix}}_{\mathbf{u}} = \underbrace{\begin{bmatrix} 0 \\ f(t) \end{bmatrix}}_{\mathbf{f}}.$$

i.e.

$$\frac{d\mathbf{u}}{dt} + \mathbf{A}\mathbf{u} = \mathbf{f}$$

Next time: will talk about the characterization of 2×2 first order systems.

Energy conservation

Velocity $\mathbf{v} = d\mathbf{u}/dt$.

$$\text{Kinetic energy: } \frac{1}{2}m_1 v_1^2 + \cdots + \frac{1}{2}m_n v_n^2 = \frac{1}{2}\left(\frac{d\mathbf{u}}{dt}\right)^T \mathbf{M}\left(\frac{d\mathbf{u}}{dt}\right)$$

Potential energy in a spring is $\frac{1}{2}c_i e_i^2$, with spring constant c_i and elongation e_i. Sum over all yields $\frac{1}{2}\mathbf{e}^T\mathbf{C}\mathbf{e}$. With $\mathbf{e} = \mathbf{A}\mathbf{u}$, we have

$$\text{Potential energy: } \frac{1}{2}\mathbf{u}^T\mathbf{K}\mathbf{u}.$$

Without any damping, or any forcing, i.e. for a system descried by the equation

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{K}\mathbf{u} = 0$$

the total energy, kinetic+potential is conserved:

$$\frac{1}{2}\left(\frac{d\mathbf{u}}{dt}\right)^T \mathbf{M}\left(\frac{d\mathbf{u}}{dt}\right) + \frac{1}{2}\mathbf{u}^T\mathbf{K}\mathbf{u} = const$$

With damping: dissipative process that converts mechanical energy into heat. Quantity above no longer conserved.

Numerical stability

Consider the first order model equation:

$$u_t = \lambda u, \quad u(0) = u_0,$$

with solution $u = u_0 e^{\lambda t}$.
Analytic stability: (solution will stay bounded) if $Re(\lambda) \leq 0$.

Discretize by a numerical method. Introduce a timestep Δt, and let u_n be the numerical approximation to $u(t_n) = u(n\Delta t)$.
We get

$$u^n = [G(\Delta t\lambda)]^n u_0, \quad n = 1, 2, \ldots$$

Numerical stability: u_n will be bounded for any n if $|G| \leq 1$.
This G will be different depending on the discretization.
Simple examples: Forward and backward Euler. (Repetition, known to you from any basic course in numerical analysis). [NOTES].
Similar analysis for linear systems of first order ODEs: Consider the eigenvalues of the system. For non-linear autonomous systems: Local analysis possible by linearization around a point.

Numerical accuracy

When applying a numerical method to a problem, a minimum requirement is that it is numerically stable.

However, we also want it to be accurate. Forward and backward Euler only first order accurate, $O(\Delta t)$. Reduce the time step by a factor of 2, and the error will be reduced by a factor of 2.

Higher order methods can be constructed. Matlab's routines `ode23` and `ode45` are time steppers that are based on explicit Runge-Kutta methods. Two methods of different order (2nd and 3rd, or 4th and 5th) are used together, to compute an estimate of the error. The size of the time step is adjusted to meet a set error tolerance, i.e. *adaptive time stepping* is used.

For certain types of problems, an advantage if discretization methods are *symplectic*. See example in Strang on p 113.