Project assignments in Engineering Mathematics.

25 november 2018

1. The output, $x(t)$, of a linear time-dependent system is given by

$$x''(t) + 2\,x'(t) + 26\,x(t) = 10\cos(t)\,H(t-\pi), \quad x(0) = \frac{1}{2}, \; x'(0) = 1,$$

where the right-hand-side of the equation represents the system input $u(t)$.
Note that the input is a bounded function. H is the Heaviside step function, *heaviside()* in Matlab.

   (a) Find the output $x(t)$ using the Matlab function `dsolve()`. This requires you to use the symbolic toolbox (*doc syms*).

   (b) Obtain $x(t)$ using Matlab functions for Laplace transform, (ordinary) equation solve, and inverse Laplace transform. Compare the results.

   (c) Produce a graphic representation/plot of $x(t)$. Make sure that the plot accounts for the entire structure of the output function.
   Is the plot consistent with the stability properties of the system and the given input?

   (d) (*Something extra for the interested.*) Obtain the solution in a purely numerical way, using Matlab's `ode45()` function. Compare the time the solution takes with `dsolve()` using *tic* to start clock and *toc* to stop it.

2. Consider the function table below

x	1	2	3	4	5	6
$f(x)$	2	3	1	5	4	1

   (a) Estimate $f'(4)$ using the forward, backward, and centered difference method with $h = 1$.

   (b) Estimate $f'(4)$ using second order interpolation polynomial. Compare with your findings above.

   (c) Estimate $\displaystyle\int_1^6 f(x)\,dx$ using the Trapezoidal rule with $h = 1$.
   Try to implement the scheme in Matlab using clean vector operations and a program utilizing a **for** loop as well. Compare with the Matlab built-in function `trapz()`.

3. Find a solution for the system of non-linear equations $\begin{cases} x_1^3 + x_1 x_2 + x_2^3 = 2 \\ x_1^3 - x_1^5 x_2 + x_2^3 = 3 \end{cases}$
by implementing Newton's method in a Matlab script.

   (a) Solve the system starting at the point (1,0) and iterate using a **for** or a **while** loop until convergence. Document/Print the current point during the procedure. There are two solutions (plot!) to this system of equations. Can you also find the other one by using a different starting point?

   (b) Compare your findings above with Matlab built-in function `fsolve()`.

4. A **time series** is really just another word for a random process with time as the discrete index space, i.e. it is a sequence of random variables $\{X_t : t \in \mathbb{Z}\}$ all defined on the same sample space. Time series are commonly recursively represented which facilitates their treatment and analysis. One way of characterizing their nature is to say that they are an *linear filtration of white noise.* Therefore we begin by defining this concept.

   Definition 1 **White noise**, *$\{\epsilon_t : t \in \mathbb{Z}\}$, is a sequence of independent standard normally distributed random variables. An* ***AR(p)* process** *is a time series, $\{X_t : t \in \mathbb{Z}\}$, which satisfies the feedback equation*

$$X_t \;=\; \alpha_1 X_{t-1} + \alpha_2 X_{t-2} + \ldots + \alpha_p X_{t-p} + \sigma_\epsilon \epsilon_t$$

   for all $t \in \mathbb{Z}$, where $\{\epsilon_t : t \in \mathbb{Z}\}$ is white noise and $\alpha_1, \alpha_2, \ldots, \alpha_p, \sigma_\epsilon$ are constants $\in \mathbb{R}$.

   Procedure 1 (Simulation of an $AR(1)$ process) *Be aware that you will need the* System Identification Toolbox *for* `ar()`*, you need to tick this when installing Matlab.*

   - *Initialization:*
     - *Assign a value to α*
     - *Let $\sigma_\epsilon = \sqrt{1-\alpha^2}$*
   - *Generate $x_1 \in N(0,1)$*
     (Hint: Use the Matlab function `randn()` *for this.)*
   - *Recursively for $t = 2, 3, \ldots, 500$*
     - *generate $\epsilon_t \in N(0,1)$*
     - *let $x_t = \alpha x_{t-1} + \sigma_\epsilon \epsilon_t$*
   - *Plot the simulated $\{x_t : t \in 1, 2, 3, \ldots, 500\}$*
     (Hint: Use the Matlab function `plot()` *and* `subplot()` *for this.)*

   (a) Do all steps of this procedure also for $\alpha = 0.5, 0.9, 0.95, 0.99$.

   (b) Compare these different simulations with each other.
   What does increasing α imply for the $AR(1)$ process?

   (c) Estimate the coefficients α and σ_ϵ from the simulated sample $\{x_t : t \in 1, 2, 3, \ldots, 500\}$.
   Compare your fabricated α values with what Matlab gives you for each increasing iteration.
   Please not that Matlab's `ar()` gives an object, the α values are addressed as *object.a*.