25 February 2025

Problem Set 2 : General Properties of Nonlinear Systems

1 Picard Approach to the Harmonic Oscillator

We solve the harmonic oscillator equation $\ddot{x} + \omega^2 x = 0$ with initial conditions $x(0) = c_0$ and $x'(0) = \omega s_0$.

(a) Rewrite the harmonic oscillator equation in the $\dot{\vec{x}}(t) = \vec{f}(\vec{x}(t))$ form and analytically compute the first two Picard iterations $\vec{x}_1(t)$ and $\vec{x}_2(t)$. Do you see where the iterations are going ? Otherwise continue with $\vec{x}_3(t)$ and so on to guess $\lim_{k\to\infty} \vec{x}_k(t)$.

(b) Prove rigorously that Picard's iterations converge to the result you guessed in (a).

<u>*Reminder*</u> : For a system $\dot{\vec{x}}(t) = \vec{f}(\vec{x}(t))$ Picard's method states that $\vec{x}_0(t) = \vec{x}(t_0)$ and $\vec{x}_{k+1}(t) = \vec{x}(t_0) + \int_{t_0}^{t} \vec{f}(s, \vec{x}_k(s))\mathrm{d}s$

2 Numerical Implementation of Picard Iteration

It is possible to use Picard iteration method to solve differential equations numerically. Write a Matlab program that numerically approximates the first N Picard iterations on a set of discrete times $t_i = t_0 + i\Delta t,\ i \in \mathbb{N}$ of an interval $[t_0, t_f]$. Start only with a first order differential equation $\dot{x}(t) = f(x(t))$ and consider $f(x) = \cos(x) + 1.1$, with $t_0 = 1$, $t_f = 51$ and $x_0 = 1/2$.

(a) Start by computing the first Picard iteration $x_1(t)$. Initialise all the necessary parameters, i.e. the discretised time vector $\vec{t} = (t_0, t_0 + \Delta t, \ldots, t_f)$, the zeroth Picard interation, which is constant $x_0(t_i) = x_0, \forall i$, and so on. The next iteration is $x_1(t) = x_0 + \int_{t_0}^{t} f(x_0(s))\mathrm{d}s$, therefore numerical integration is needed. This integral needs to be computed for all the discrete values of time $t = t_i$. To limit redundant computations and gain in speed, when evaluating $\int_{t_0}^{t_{i+1}} f(x_k(s))\mathrm{d}s$ you can reuse the integral $\int_{t_0}^{t_i} f(x_k(s))\mathrm{d}s$ computed for $x_{k+1}(t_i)$ and add the next bit $\int_{t_i}^{t_{i+1}} f(x_k(s))\mathrm{d}s$. We suggest to use the trapezoidal rule. Verify that $x_1(t)$ is a straight line.

(b) Now, just repeat the process to get the N^{th} Picard iteration and show visually that the iterations converge to x_{sol}, the solution of the differential equation computed with `ode45`.

(c) The norm $||x_a - x_b||_2 = \sqrt{\int_{t_0}^{t_f} [x_a(t) - x_b(t)]^2 \mathrm{d}t}$ can be used to measure the distance between two functions. For $t_f = 11$ and $\Delta t = 10^{-1}$ plot the distance with the final solution $||x_n - x_{\mathrm{sol}}||_2$ as a function of n (ranging from $n = 0$ to 20). Use the default relative error tolerance `rel_tol=1e-3`. Identify what causes the error to plateau for high-n Picard iterations.

<u>*Tip*</u> : Instead of giving `ode45` only the initial and final time `[t0,tf]` and let it choose at which times it evaluates the solution, you can give `ode45` the time vector $\vec{t}$ where the solutions should be evaluated. Also, to set the relative error tolerance, use `ode45(@f,t,x0,options)` with `options = odeset('RelTol',1e-5)`.

(d) Plot again $||x_n - x_{\mathrm{sol}}||_2$ as a funtion of n to compare on the same plot the time steps $\Delta t = 0.1$, 0.01, 0.001 and 0.0001. Make the plot for x_{sol} evaluated with `rel_tol=1e-3`, `1e-5` and `1e-7`. Explain the results.

BONUS

(e) If your code is well organized, you can try with only a few changes to compute the Picard iterations for a system of arbitrary dimension !

3 Lipschitz Constant

Find a Lipschitz constant, if it exists, for the following functions in the indicated domains.
Reminder : A Lipschitz constant, K, on the domain D is a constant such that $||\vec{f}(\vec{x}) - \vec{f}(\vec{y})||_2 < K||\vec{x} - \vec{y}||_2$, $\forall \vec{x}, \vec{y} \in D$, where $|| \cdot ||_2$ is the standard Euclidean norm.

(a) $f(x) = \cos(\omega x)$, $x \in [-\pi, \pi]$

(b) $f(x) = \sqrt[3]{x}$, $x \in [-1, 1]$

(c) $f(x_1, x_2) = \frac{x_1 x_2}{1+x_1^2+x_2^2}$, $x_1^2 + x_2^2 \leq 16$