

The SS2INI Program

1 Initialization and the Swinging Spring

SS2INI is a MATLAB program to solve the equations for the elastic pendulum or Swinging Spring in two dimensions. This system comprises a heavy bob suspended by a light elastic spring. The bob is free to move in a vertical plane.

The procedure of linear and nonlinear normal mode initialization can be clearly illustrated by applying the method to the equations of the swinging spring. The dynamical equations are derived in this section and the processes of linear and nonlinear initialization are outlined. The MATLAB program is described in §2. A numerical example and sample output are included. A few suggestions for experiments are given in §3.

1.1 The Dynamical Equations

Let ℓ_0 be the unstretched length of the spring, k its elasticity or stiffness and m the mass of the bob. At equilibrium the elastic restoring force is balanced by the weight: $k(\ell - \ell_0) = mg$. Polar coordinates $q_r = r$ and $q_\theta = \theta$ are used, and the radial and angular momenta are $p_r = m\dot{r}$ and $p_\theta = mr^2\dot{\theta}$. The Hamiltonian is, in this case, the sum of kinetic, elastic potential and gravitational potential energy:

$$H = \frac{1}{2m} \left(p_r^2 + \frac{p_\theta^2}{r^2} \right) + \frac{1}{2}k(r - \ell_0)^2 - mgr \cos \theta.$$

The (canonical) dynamical equations may now be written explicitly

$$\begin{aligned} \dot{\theta} &= p_\theta / mr^2 \\ \dot{p}_\theta &= -mgr \sin \theta \\ \dot{r} &= p_r / m \\ \dot{p}_r &= p_\theta^2 / mr^3 - k(r - \ell_0) + mg \cos \theta. \end{aligned}$$

These equations may also be written symbolically in vector form

$$\dot{\mathbf{X}} + \mathbf{L}\mathbf{X} + \mathbf{N}(\mathbf{X}) = 0$$

where $\mathbf{X} = (\theta, p_\theta, r, p_r)^T$, \mathbf{L} is the matrix of coefficients of the linear terms and \mathbf{N} is a nonlinear vector function.

Let us now suppose that the amplitude of the motion is small, so that $|r'| = |r - \ell| \ll \ell$ and $|\theta| \ll 1$. The state vector \mathbf{X} is comprised of two sub-vectors:

$$\mathbf{X} = \begin{pmatrix} \mathbf{Y} \\ \mathbf{Z} \end{pmatrix}, \quad \text{where } \mathbf{Y} = \begin{pmatrix} \theta \\ p_\theta \end{pmatrix} \quad \text{and} \quad \mathbf{Z} = \begin{pmatrix} r' \\ p_r \end{pmatrix},$$

and the linear dynamics of these components evolve independently. We call the motion described by \mathbf{Y} the rotational component and that described by \mathbf{Z} the

elastic component. The rotational equations may be written

$$\ddot{\theta} + (g/\ell)\theta = 0$$

which is the equation for a simple pendulum having oscillatory solutions with frequency $\sqrt{g/\ell}$. The remaining two equations yield

$$\ddot{r} + (k/m)r = 0,$$

the equations for elastic oscillations with frequency $\sqrt{k/m}$. We define the rotational and elastic frequencies and their ratio by

$$\omega_R = \sqrt{\frac{g}{\ell}}, \quad \omega_E = \sqrt{\frac{k}{m}}, \quad \epsilon \equiv \left(\frac{\omega_R}{\omega_E}\right).$$

It is easily shown that the rotational frequency is always less than the elastic:

$$\epsilon^2 = \frac{mg}{k\ell} = \left(1 - \frac{\ell_0}{\ell}\right) < 1, \quad \text{so that} \quad |\omega_R| < |\omega_E|.$$

We assume that the parameters are such that $\epsilon \ll 1$. In this case the linear normal modes are clearly distinct: the rotational mode has low frequency (LF) and the elastic mode has high frequency (HF).

1.2 Linear and Nonlinear Initialization

For small amplitude motions, for which the nonlinear terms are negligible, the LF and HF oscillations are completely independent of each other and evolve without interaction. We can suppress the HF component completely by setting its initial amplitude to zero:

$$\mathbf{Z} = (r', p_r)^T = \mathbf{0} \quad \text{at} \quad t = 0.$$

This procedure is called linear initialization. When the amplitude is large, nonlinear terms are no longer negligible and the LF and HF motions interact. It is clear from the equations that linear initialization will not ensure permanent absence of HF motions: the nonlinear LF terms generate radial momentum. To achieve better results, we set the initial *tendency* of the HF components to zero:

$$\dot{\mathbf{Z}} = (\dot{r}, \dot{p}_r)^T = \mathbf{0} \quad \text{at} \quad t = 0,$$

This procedure is called nonlinear initialization. For the spring, we can deduce explicit expressions for the initial conditions:

$$r(0) = r_B \equiv \frac{\ell(1 - \epsilon^2(1 - \cos\theta))}{1 - (\theta/\omega_E)^2}, \quad p_r(0) = 0.$$

Thus, given arbitrary initial conditions $\mathbf{X} = (\theta, p_\theta, r, p_r)^T$, we replace $\mathbf{Z} = (r, p_r)^T$ by $\mathbf{Z}_B = (r_B, 0)^T$. The rotational component $\mathbf{Y} = (\theta, p_\theta)^T$ remains unchanged.

2 Description of the SS2INI Program

2.1 Parameter Setup

The basic spring parameters are given default values:

- m : Mass of bob (1 kg)
- g : Acceleration of gravity (π^2 m²/s²)
- k : Stiffness of spring ($100\pi^2$ N/m)
- l : Length at equilibrium (1 m).

Additional constants are then calculated:

$$\epsilon = \sqrt{\frac{mg}{k\ell}}, \quad \ell_0 = (1 - \epsilon^2)\ell, \quad \omega_R = \sqrt{\frac{g}{l}}, \quad \omega_E = \sqrt{\frac{k}{m}}, \quad \tau_R = \frac{2\pi}{\omega_R}, \quad \tau_E = \frac{2\pi}{\omega_E}.$$

For the default parameters, $\epsilon = 1/10$, $\tau_R = 2$ s and $\tau_E = 0.2$ s.

There are two passes through the program, First for data which is linearly initialized, second for data which is nonlinearly initialized (LOOP = 1:2).

The initial conditions have the following default values (LNMI):

$$\theta = 1.0 \quad \dot{\theta} = 0 \quad r = l \quad \dot{r} = 0$$

For NNMI, the value of r is defined (as explained above) as

$$r = \frac{\ell(1 - \epsilon^2(1 - \cos\theta))}{1 - (\dot{\theta}/\omega_E)^2}$$

The length of the integration (in seconds) is determined by `tmax`. The default value is 6 seconds.

2.2 Solution of the Equations

The central section of the program contains in-line code to solve the four ordinary differential equations. Normally, code between the lines marked 'VVVV' and 'AAAA' should not be changed! If you want to experiment, save a copy of the program.

The initial positions and momenta are put in the vector x_0 . The numerical integration is then handled by the method of ODE23. To avoid function calls, the code is placed in-line. It uses second and third order Runge-Kutta-Fehlberg integration methods with an automatic adjustment of step-size to meet a specified tolerance. Since the method uses an adaptive time step, the results are generated on an irregular grid. For simplicity, the solution on a regular time grid `treg` is calculated, and stored in `xreg`. By default, $nt = 512$ values are calculated. The total energy at each time is calculated:

$$E = \frac{1}{2}m(\dot{r}^2 + (r\dot{\theta})^2) + \frac{1}{2}k(r - \ell_0)^2 - mgr \cos\theta - E_0$$

where $E_0 = \frac{1}{2}k(\ell - \ell_0)^2 - mg\ell$.

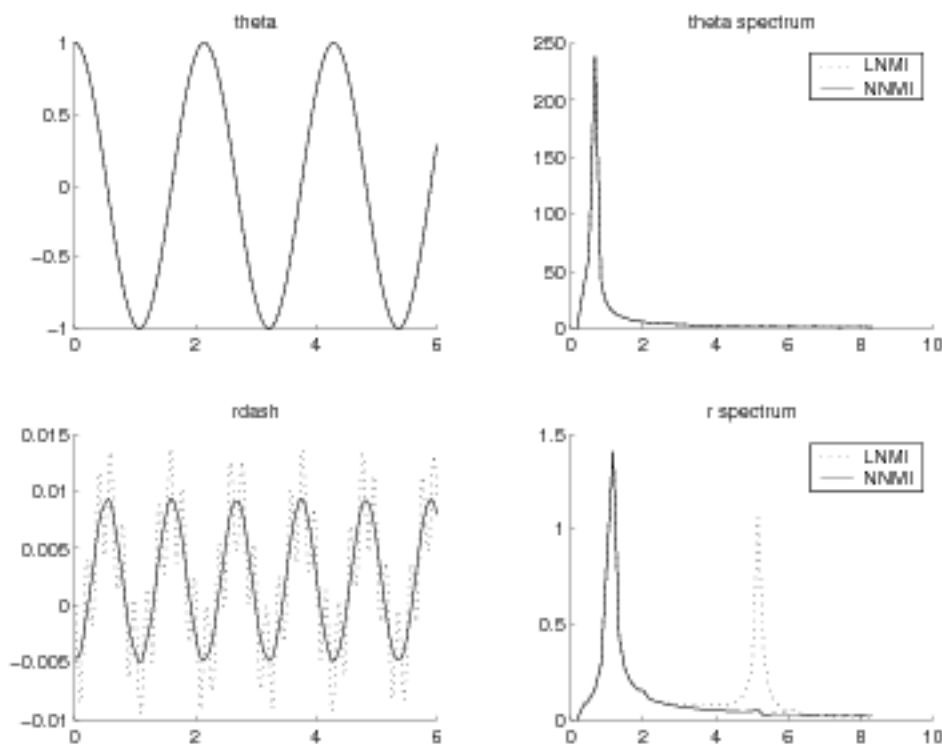


Figure 1: Solution of swinging spring equations for linear (LNMI) and nonlinear (NNMI) initialization. See text for details.

2.3 Plotting the Results

The default output comprises a figure with four panels. For each panel there are two graphs, the dotted red curve is for linearly initialized data (LNMI). The solid blue curve is for nonlinearly initialized data (NNMI).

- In the upper left panel `subplot(2,2,1)` the 'slow' variable θ is plotted as a function of time.
- In the upper right panel `subplot(2,2,2)` the frequency spectrum of θ is shown. The abscissa is the frequency in cycles per second.
- In the lower left panel `subplot(2,2,3)` the 'fast' variable $r' = r - \ell$ is shown against time.
- In the lower right panel `subplot(2,2,4)` the frequency spectrum of r' is shown (against frequency in cycles per second).

2.4 A Numerical Example

In Figure 1 we show the results of two integrations of the spring equations. The upper panels show the evolution and spectrum of the slow variable θ ; the lower panels are for the fast variable r . Dotted curves are for linear initialization and solid curves for nonlinear initialization. The parameter values are $m = 1$, $g = \pi^2$, $k = 100\pi^2$ and $\ell = 1$ (all SI units), $m = 1$, $g = \pi^2$, $k = 100\pi^2$ and $\ell = 1$ (all SI units), so that $\epsilon = 0.1$ and the periods of the swinging and springing motions are respectively $\tau_R = 2\text{s}$ and $\tau_E = 0.2\text{s}$. The initial conditions are vanishing velocity ($\dot{\theta} = \dot{r} = 0$), with $\theta(0) = 1$ and $r(0) \in \{1, 0.99540\}$. The equations are integrated over a period of 6 seconds. For the slow variable, the curves are indistinguishable. The spectrum has a clear peak at a frequency of 0.5 cycles per second. For the fast variable, the linearly initialized evolution has high frequency noise (dotted curve, lower left panel). This is confirmed in the spectrum: there is a sharp peak at 5 cps. When nonlinearly initialized, this peak is removed: only the peak at 1 cps remains. This is the *balanced fast motion*. It can be understood physically: the centrifugal effect stretches the spring twice for each pendular swing: the result is a component of r with a period of one second. The radial variation does not disappear for balanced motion, but it is of low frequency!

3 Suggested Exercises

- Run the program with the default settings and evaluate the results.
- Vary the spring stiffness parameter k (default $k = 100\pi^2$) to $k = 400\pi^2$ and consequently the frequency of the ‘fast’ motion. The frequency ratio changes to $\epsilon = 1/20$. Examine the plots.
- Reduce the spring stiffness parameter k to make the ratio smaller (say $\epsilon = 1/4$). Separation between slow and fast motion is more fuzzy. How does the initialization perform?
- The *Resonant Case* $\epsilon = 1/2$ is particularly interesting. Select the stiffness k to achieve this. Energy flows between the slow and fast motions with a long period. Integrate over a longer time `tmax=50` and study the solution.
- Set initial conditions which are far from balance. (For example, set $r = 1.5$). Disable the initializations. How does the integration look? Apply LNMI and NNMI and study the difference.
- DFI: Integrate over a span $T = 2\text{s}$ and apply a digital filter to the time series solutions. (The simplest filter is a straight time-averaging). Restart from the filtered data. How does the integration look?