Teaching Wave Mechanics with Matlab

Enrico Onofri

Dipartimento di Fisica e Scienze della Terra
Università di Parma
INFN Sezione di Milano Bicocca
Gruppo Collegato di Parma

Abstract

I'll present several programs which I developed along the years which allow to compute energy spectra, tunneling amplitudes, animations of Schroedinger wave packets in various potentials in one and two dimensions, perturbation theory to high orders, Clebsch-Gordan, and more. All very elementary, but useful in introductory QM courses to explore the properties of simple quantum systems. I'll add some comments on another kind of programs dealing with fundamental problems in classical mechanics, like (not-so)elementary celestial mechanics and synchronization of Huygens pendulums. Emphasis is given on programming style and the use of efficient algorithms, notably "fast" algorithms such as sparse matrix computations, fast transforms, efficient ODE solvers with event capturing. All the programs are available on the author's web site or on the cloud.

Introduction

I have decided to put in the foreground the applications relevant to teaching Quantum Mechanics at an introductory level, hoping that this will be useful to a few colleagues. Obviously there are other examples of numerical and symbolic computations of this kind on the market, but this is what I developed in the course of many years of teaching introductory QM and I believe that they may raise some interest . Several other programs about classical mechanics are just outlined in the second part, among these Mercury perihelion shift and synchronization of cuckoo clocks. I am indebted to many colleagues for collaboration and inspiration, in a special way to *G.P. Tecchiolli, C. Destri, F. Di Renzo and G. Cicuta.*

Sections

1. Quantum Mechanics
2. Classical Mechanics
3. Miscellanea

Quantum Mechanics

The main tools in my approach to the problems posed by QM are

- *"sparse matrix" techniques*
- *fast transforms*
- *efficient ODE (ordinary differential equations) solvers.*

All this is implemented in `matlab` *which is my language of choice both for programming simplicity, easy plotting routines, fast prototyping, speed in execution, the only shortcoming being the fact that it is not free software. Some younger colleagues are suggesting* python *as a free software alternative since many years, but I cannot learn one more language now, after Ftn, C, perl, matlab, form, mathematica… I'll leave them the pleasure to develop a new suite of scientific programs in the new language. The ideas can be exported rater easily.*
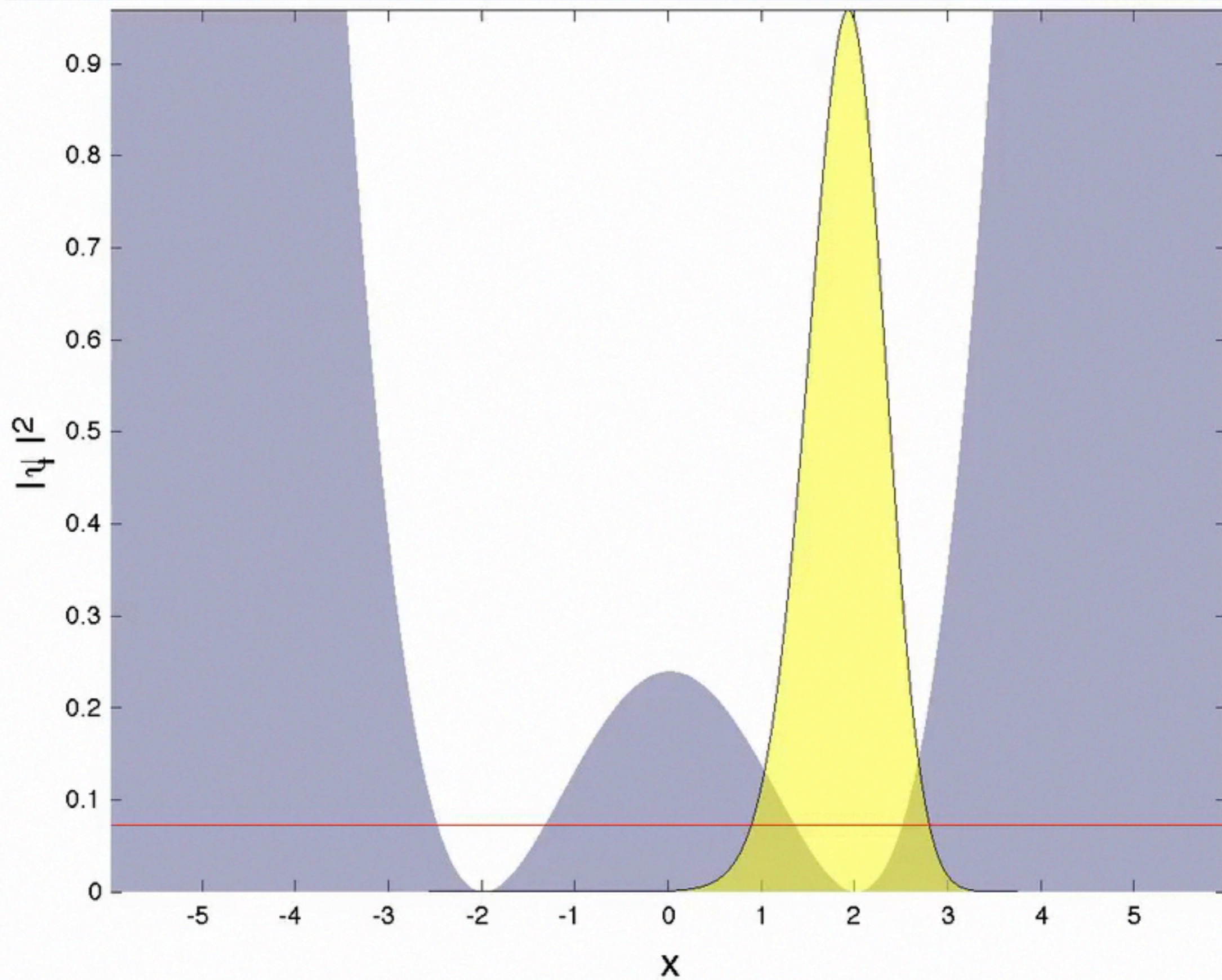
*1. Wave function evolution for Schroedinger equation. The most efficient technique I know combines **Lie-Trotter** formula (path-integral in phase space) with the Fast Fourier Transform. The program initially computes two vectors which contain the values of $exp(-½i\,\tau\,p^2)$ and $exp(-i\,\tau\,V(x))$ defined on a finite grid in configuration space and in momentum space. A single evolution step is given by*

$$\psi(x) \Rightarrow \mathcal{F}^{-1} \exp\{-\tfrac{1}{2}i\tau p^2\}\, \mathcal{F} \exp\{-i\tau\, V(x)\}\, \psi(x)$$
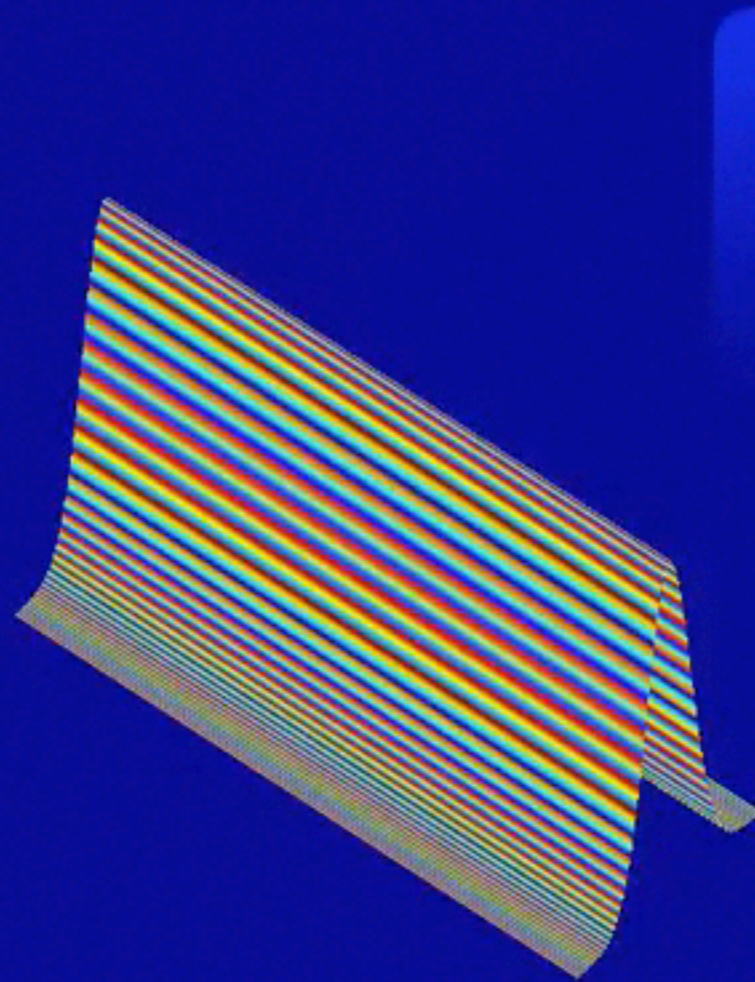
*where $\mathcal{F}$ is the Fast Fourier Transform (FFT). This operation is then iterated N-times to cover the whole time interval required. At every step the probability density $|\psi(x)|^2$ or $|\varphi(p)|^2$ or both can be visualized. **A nice feature of the Trotter formula is that it preserves unitarity and time reversal invariance to machine precision.** Also time-dependent potentials are feasible in this scheme, with the only overload of computing $exp(-i\,\tau\,V(x,t))$ at each step.*

*Examples are given in ~/**MeccanicaQuantistica/WMS**, where ~ is Cortona14 on* dropbox *or on the author's web site. See the code doublewell.m (motion in the potential with two degenerate classical minima) and wms2.m (wave packet motion in 2-dim with various potential barriers). The same program wms2 can be run in "W" mode where the wave equation (no-dispersion) is studied for contrast to dispersive de Broglie waves. Video clips are available on dropbox and my web site.*

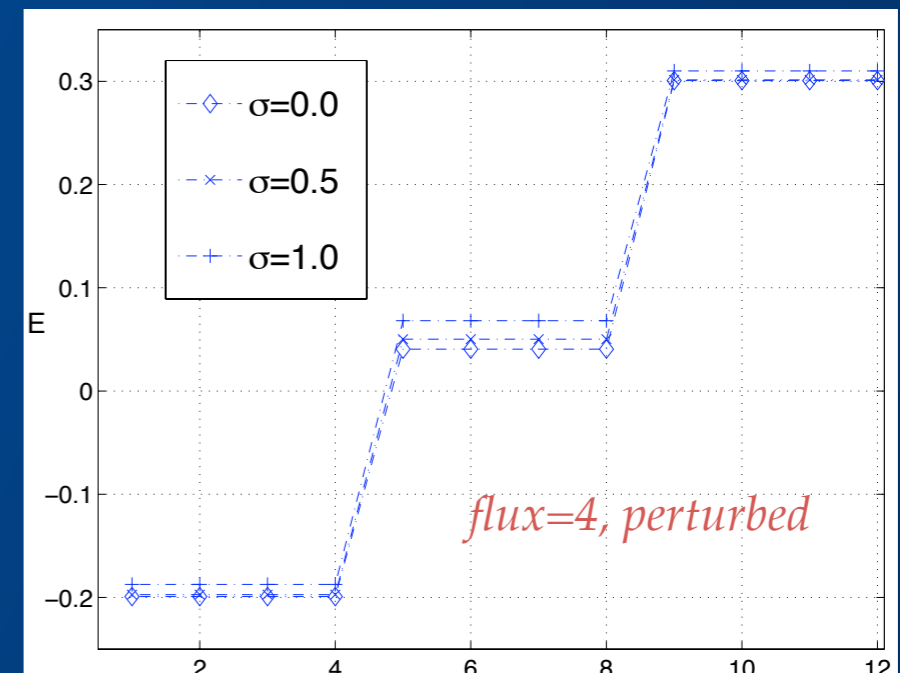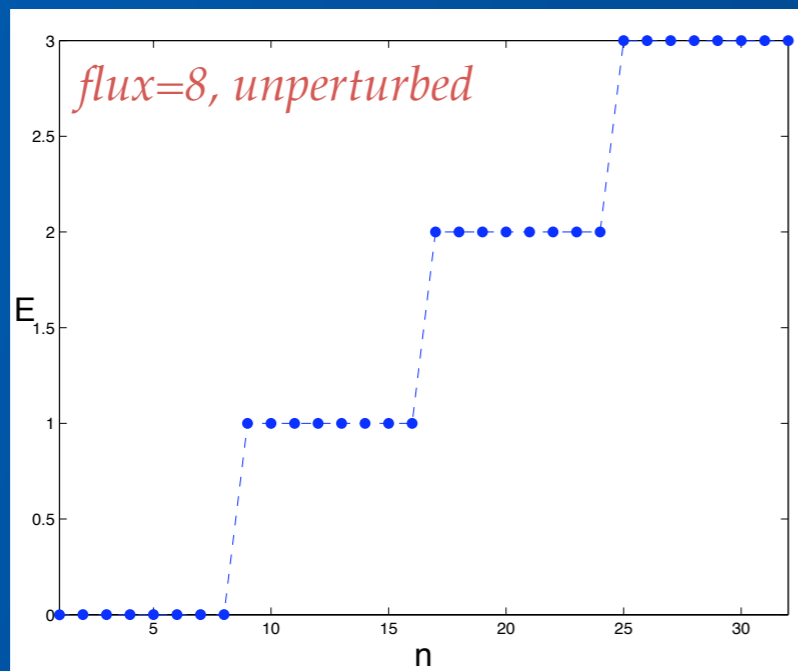Schroedinger Wave packet evolution in a double-well potential

2. Energy spectrum calculation for Hamiltonians in 1-2-3 d.o.f.:
~/MeccanicaQuantistica/spectra.

*One exploits the flexibility of the routine "eigs" which incorporates the ARPACK suite. This allows either to use sparse matrices or to directly **define the action of the Hamiltonian through an external routine**; this has the advantage of being allowed to use rather big vectors without worrying about memory exhaustion by huge matrices - think of a 2-dim wave function sampled on a 128x128 grid: the corresponding finite difference Laplacian, if not defined with care, could fill $16x128^4 \approx 4$ GBy of memory. But of course one does not use full matrices and eigs does even better. The Laplacian is defined by FFT and it costs nothing more than $128^2 \ln(128)$ both in time and memory. Try schr'n'D, with n=1,2,3. Especially in higher dimensions the programs let you explore the degeneracy of energy levels and its resolution by small perturbations. Notice that the spectral method (representing p^2 by FFT) allows for high accuracy, which is particularly welcome when one explores degenerate spectra like 3-dim potentials with high symmetry or even Landau levels.*

- *The same technique is used in the case of Schroedinger equation with a magnetic field, ~/MeccanicaQuantistica/spectra/Landau. The particle is confined to a toroidal surface with a transverse magnetic field. The algorithm takes into account the nature of the wave function which in this case is defined by a section of the vector bundle over T^2 which is the standard description for gauge interaction on closed surfaces, in particular one has to take into account the Dirac monopole quantization condition. The program lets you check the degeneracy of Landau levels - given by the integral flux - and also explore the breaking of degeneracy for deformations of the magnetic field [1]. The program uses an idea introduced [2] in 1991 to adapt the Lie-Trotter formula to the magnetic case. Here the accuracy allowed by the spectral method is essential - perturbation is first-order for degenerate states!*

flux=8, unperturbed

flux=4, perturbed

1) E. Onofri, *"Splitting Landau levels on the torus by periodic perturbations"*, *Int.J.Mod.Phys.C*, **19** No.11 (2008) 1753.
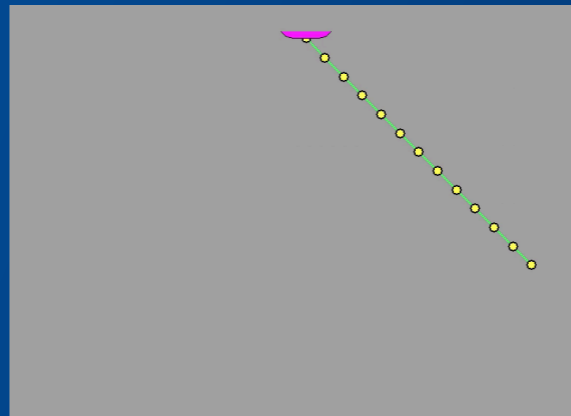2) P. Maraner, E. Onofri, G.P.Tecchiolli, *J.Comput.Appl. Math.*, **37** (1991) 209-219.

★ *The* `matlab` *routine "kron" generates Kronecker products of matrices, which allows to build creation/annihilation matrices in any dimension starting from the 1-dim bosonic matrices. With these routines one can explore the spectra of Hamiltonians built out of bosonic operators:* ~/MeccanicaQuantistica/spectra/a-adag.

★ ~/MeccanicaQuantistica/tunneling: *here you may find programs aimed at visualizing the penetration of potential barriers, the calculation of the transmission/reflection coefficients also in the case of an array of potential barriers* $\Sigma_n V(x - n L)$.

★ ~/MeccanicaQuantistica/miscellanea: *at the moment this folder contains a routine to calculate Clebsch-Gordan coefficients for the sum of 2 or 3 angular momenta. It makes use of "kron" to define the matrices* $J^{(1)} \otimes 1 + 1 \otimes J^{(2)}$ *and* $J^{(1)} \otimes 1 \otimes 1 + 1 \otimes J^{(2)} \otimes 1 + 1 \otimes 1 \otimes J^{(3)}$. *There is a subtle problem still to be solved: "eigs" does not know Wigner's phase convention…*

II. Classical Mechanics

Aside from QM there obviously are other topics in Physics teaching which can profit from numerical analysis. Here I just mention a few of them which I considered in recent years:

▷ *Celestial mechanics: planetary motion and the calculation of Mercury's perihelion*

▷ *Celestial mechanics: stability of a Lagrange satellite in L4 position between Earth and Moon* [3].

▷ *Continuum mechanics: normal modes of arbitrarily shaped membranes.*

▷ *Many-body dynamics: motion of n-point masses constrained as a chain (n-pendulum)*

▷ *Many-body dynamics: synchronization of pendulums hanging from the same support.*

3) E. Onofri, "Elementary Celestial Mechanics using matlab", Comp. in Sci. & Engin., 3 (6) 48-53 (2001).
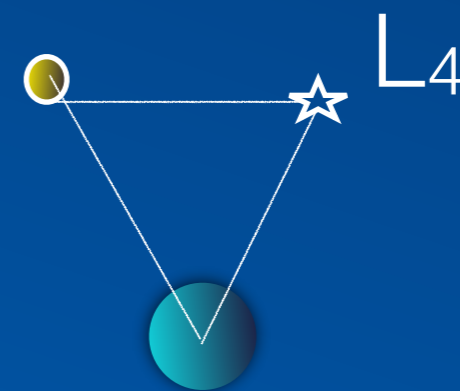
Newtonian theory would give Mercury a precession

$$\Delta\varphi_N = 5557.62 \pm 0.20'' \; (\text{☿})$$

which about 5025 due to the rotation of the earth-based astronomical coordinate system, and about 532'' is due to gravitational perturbations calculated Newtonian perturbation theory from the motion of the other planets, chiefly us, earth, and Jupiter. The precession actually observed is

$$\Delta\varphi_{\text{OBS}} = 5600.73 \pm 0.41'' \; (\text{☿})$$

the value $\Delta\varphi = 43.11 \pm 0.45''$ quoted above for the "observed" anomalous precession is obtained by subtracting the Newtonian precession from what is observed, that is,

$$\Delta\varphi = \Delta\varphi_{\text{OBS}} - \Delta\varphi_N \qquad (8.6.13)$$

My interest in Mercury's perihelion stems from a portion of Weinberg's book on Gravitation. It is natural to ask oneself: ok, I know how Einstein computed his 43"/Cy, but how can one compute the 532? In the XIXth century, 30 years of hard work using Hamilton-Jacobi perturbation methods were needed to get to the final result; without the heroic effort of Newcomb and collaborators Einstein would not have found such a strong confirmation of his theory. Today with some patience and a standard personal computer the calculation runs for a few minutes. One has to download orbital elements for the planets, organize the calculation using matlab's ode113 … et voilà $\Delta\varphi \sim 530''/Cy \pm 5''$. The value is not universal, it depends on the epoch in which you do the simulation.
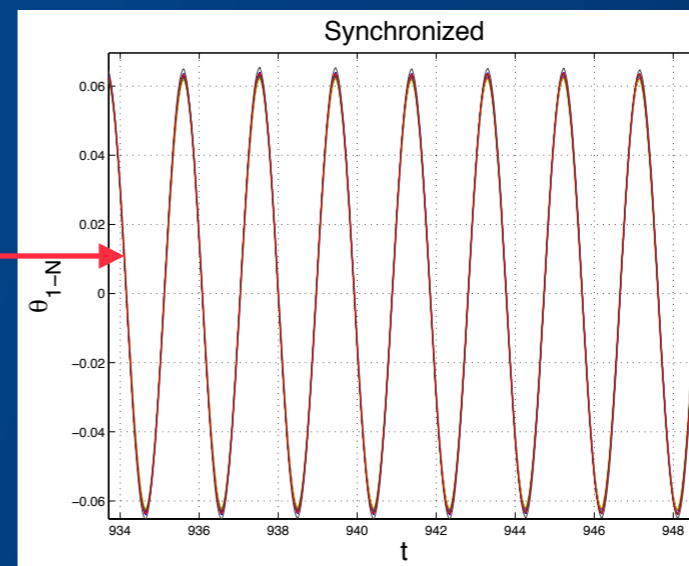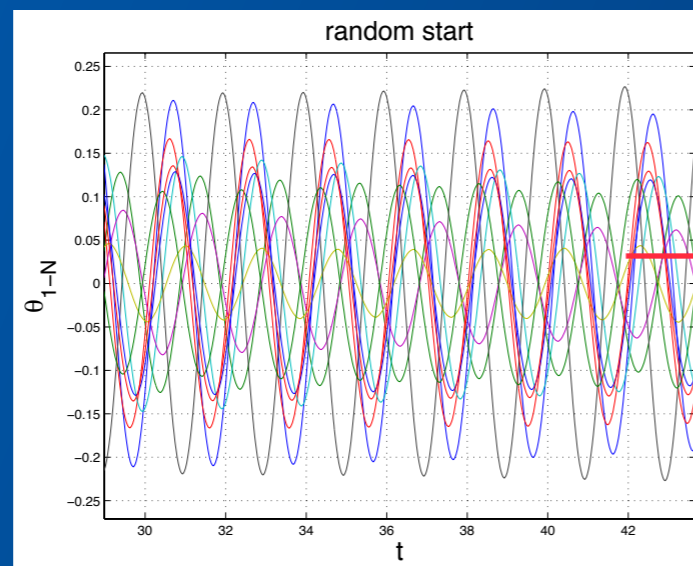
Lagrange satellite: ~/MeccanicaClassica/meccanicaceleste/MOON. It was proved by <u>Giuseppe Luigi Lagrange</u> that the motion of a satellite at position L_4 is stable. This is true under the assumption that there are just three bodies at the vertices of an equilateral triangle. But what happens if there are other bodies around? The problem Sun-Earth-Moon+L_4 was considered in Ref.3. Evidence is obtained that the perturbing action on the part of the Sun makes the Satellite unstable. At the same time the motion of the Moon can be examined, a classical problem which was very tough to solve. The program computes the variation in time of the Moon's orbit eccentricity which shows rapid oscillation on the time scale of months. Also the famous cycle of ~18 years (Saros) which is relevant for the Solar eclipses can be deduced using the program.
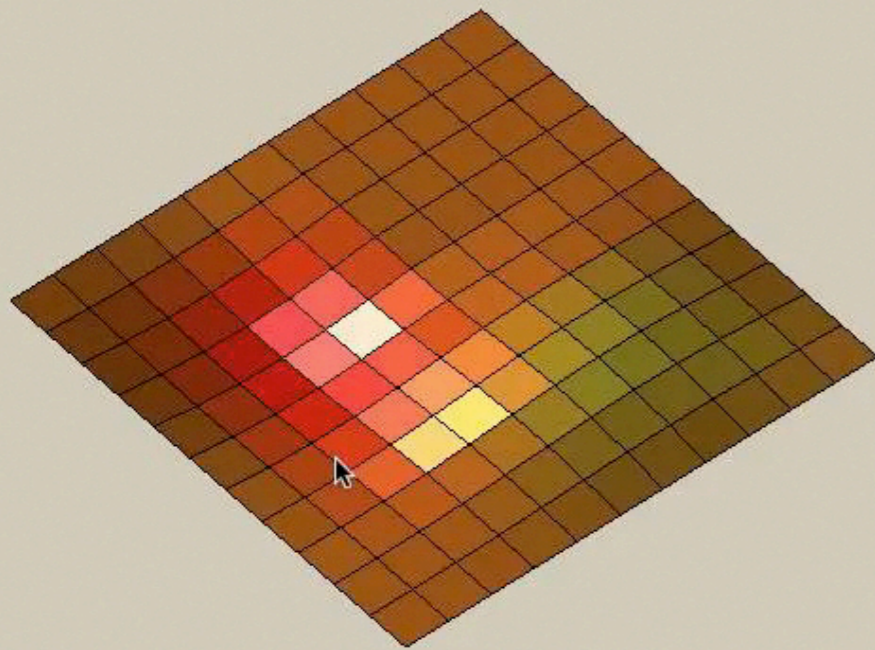
L_4

1) E. Onofri, "Elementary Celestial Mechanics using matlab", Comp. in Sci. & Engin., 3 (6) 48-53 (2001).

Last topic: clock synchronization - a classical problem studied long ago by Huygens. One can explore the problem with matlab, *taking into account that both friction and a "escapement" mechanism must be implemented [*].*

Folder: ~/MeccanicaClassica/pendoli. *One can use the "events" facility of ODE matlab routines to capture the precise instant at which each pendulum reaches the maximum amplitude; at that point a tiny impulse is applied in such a way to compensate for the energy dissipation by friction. The mechanism is quite robust: even if the kick is given with a certain probability, synchronization eventually sets in.*
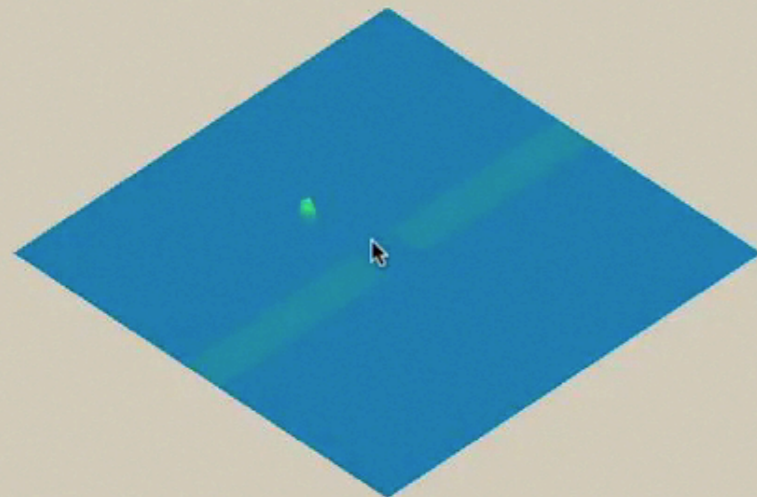

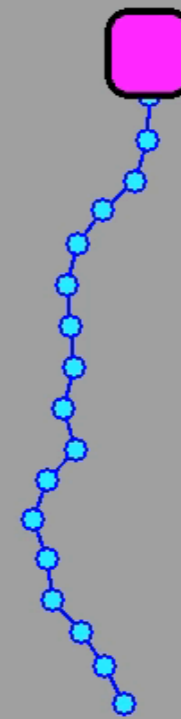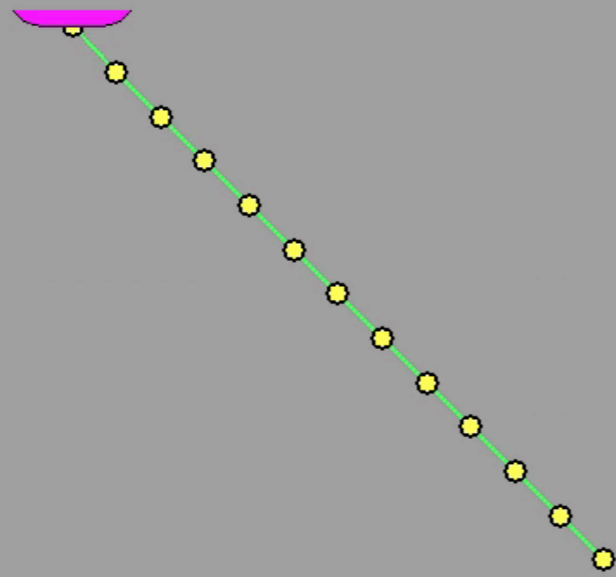*[*] I was introduced to this kind of problem by my colleague and friend Giovanni Cicuta.*
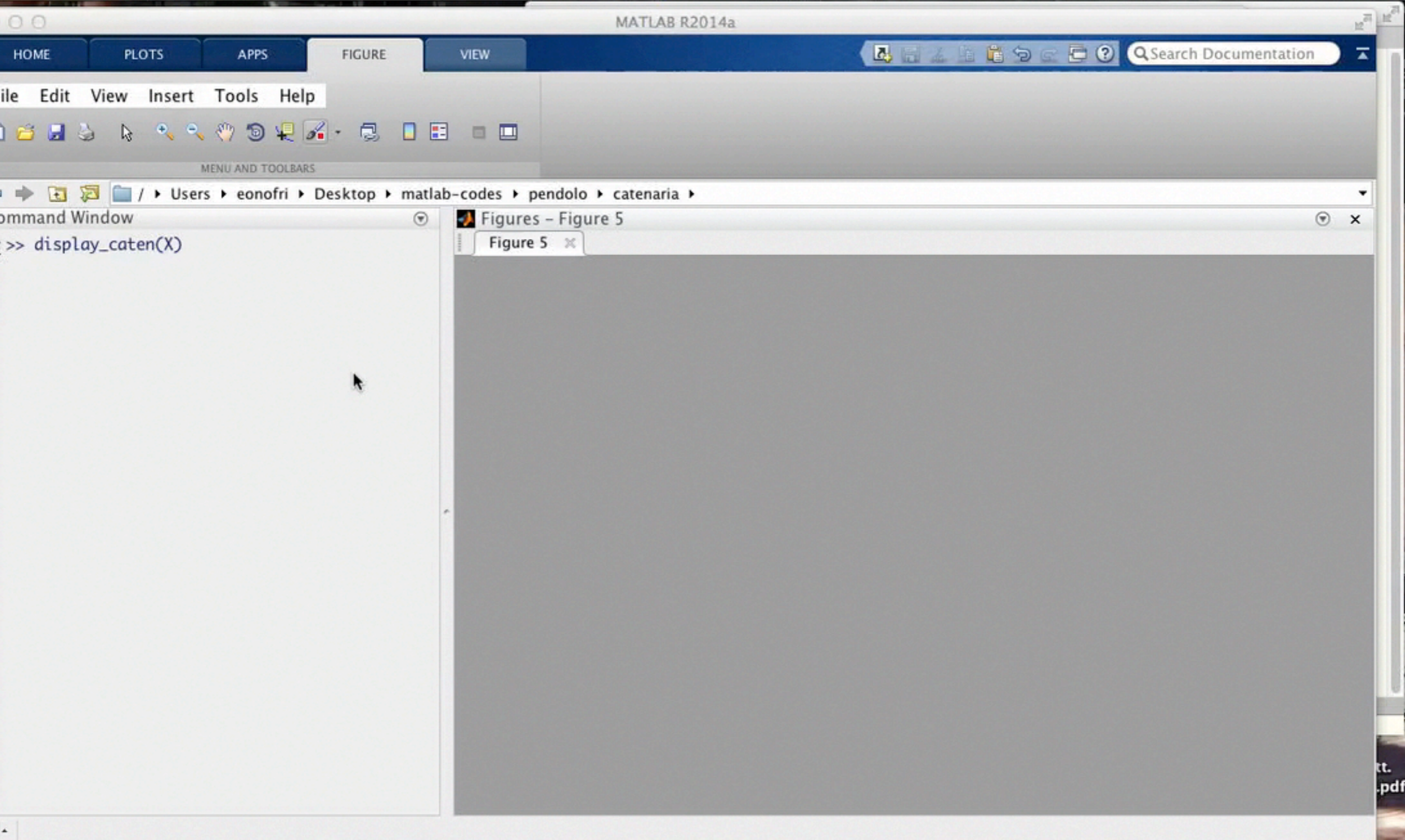
You'll find some documentation in ~/docs. To get access to the routines: go to

http://www.fis.unipr.it/~enrico.onofri/Cortona14

or send an email to me asking to be admitted to the dropbox repository - the advantage consists in automatic synchronization of files which is not the case with the web page.

Conclusions

No conclusions: the effort is going on to produce useful routines for our students … it's really fun.