Physics through Computational Thinking

The Monte Carlo method

Auditya Sharma and Ambar Jain
Dept. of Physics, IISER Bhopal

Outline

In this module we look at

1. Linear systems: Love affairs. An elementary use of the use Monte Carlo method.

General Theory of 2-d Linear Systems

The general problem is

$$\begin{aligned} \dot{x} &= a\,x + b\,y \\ \dot{y} &= c\,x + d\,y \end{aligned} \tag{1}$$

and

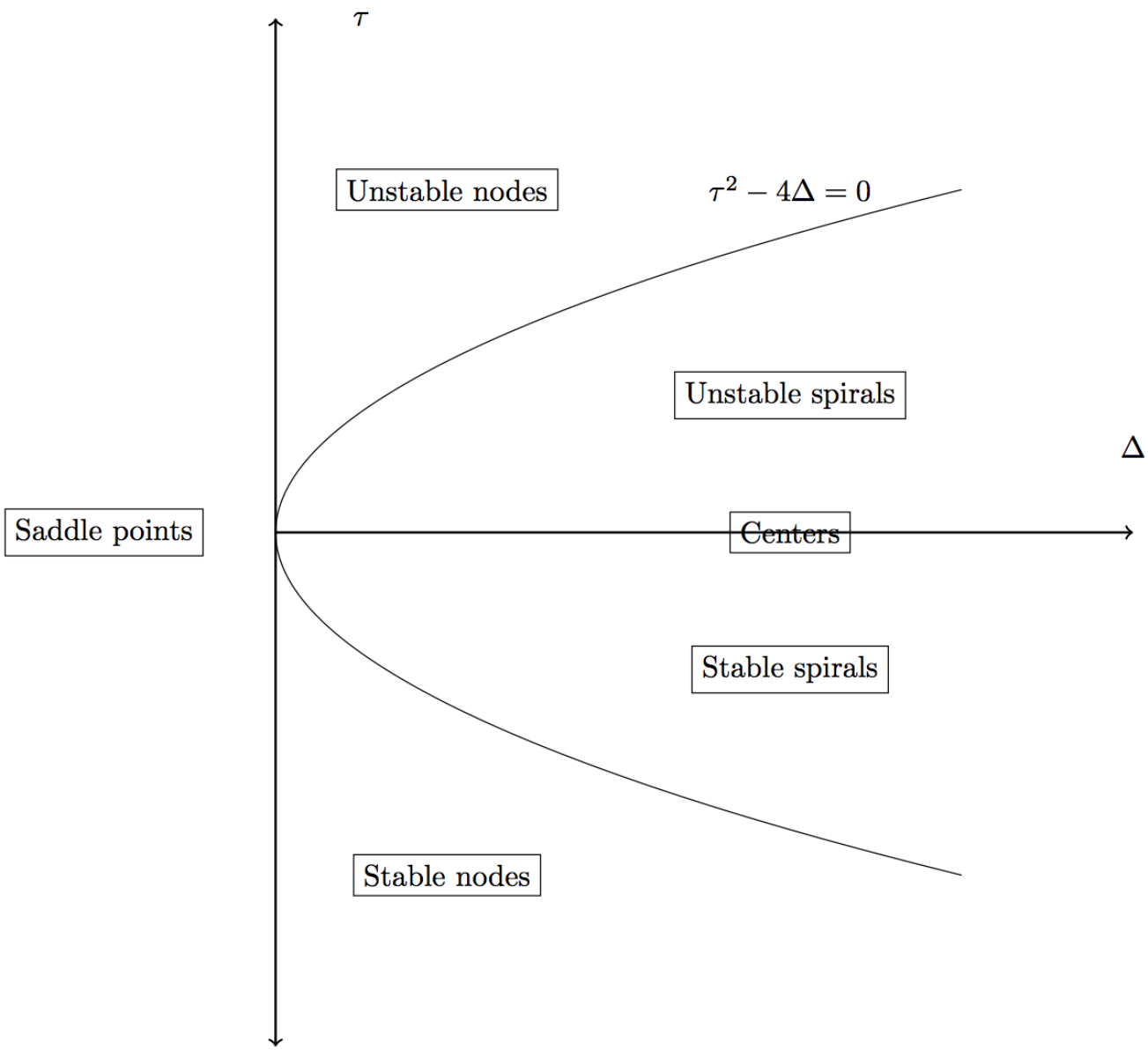
$$\lambda_1 = \frac{\tau + \sqrt{\tau^2 - 4\,\Delta}}{2} \qquad \lambda_2 = \frac{\tau - \sqrt{\tau^2 - 4\,\Delta}}{2}. \tag{2}$$

The inverse relation is

$$\tau = \lambda_1 + \lambda_2 \qquad \Delta = \lambda_1\,\lambda_2\,. \tag{3}$$

We make the following observations:

- If $\Delta < 0$, then both the eigenvalues *have to* be real, and with opposite signs. Hence the fixed point is guranteed to be a saddle point.
- If $\Delta > 0$, then we have a range of different possibilities, depending on the value of τ. If $\tau^2 > 4\,\Delta$, then the eigenvalues are real, and therefore the fixed point is a *node*. On the other hand, if $\tau^2 < 4\,\Delta$, then the eigenvalues are complex (conjugates of each other), and the fixed point then becomes either a center or a spiral.


τ
Unstable nodes
τ² − 4Δ = 0
Unstable spirals
Δ
Saddle points
Centers
Stable spirals
Stable nodes

Random systems: A first look at a Monte Carlo approach

Consider the linear system

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\,\boldsymbol{x}, \tag{4}$$

where

$$\boldsymbol{x} = \begin{pmatrix} x \\ y \end{pmatrix} \quad \boldsymbol{A} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}. \tag{5}$$

and where the numbers a, b, c, d are drawn independently and from a uniform distribution in the interval $[-1, 1]$.

- Let us generate a million such random matrices, and count the fraction of these matrices that are saddle points, unstable nodes, stable nodes, stable spirals, unstable spirals. These numbers provide excellent estimates of the probabilities of the fixed point having this nature. This approach to problem solving which involves the generation of lots of pseudo-random numbers, is an elementary example of a powerful general approach called the Monte Carlo method.

```
alist = Table[2 (RandomReal[] - 0.5), {i, 1, 10}]
```

```
{-0.276992, 0.796025, -0.893391, 0.739421, 0.231284, 0.450271, -0.486965, -0.910103, -0.711798, 0.428488}
```

Solution

```
data = Table[alist = Table[2 (RandomReal[] - 0.5), {i, 1, Nmat}];
   blist = Table[2 (RandomReal[] - 0.5), {i, 1, Nmat}];
   clist = Table[2 (RandomReal[] - 0.5), {i, 1, Nmat}];
   dlist = Table[2 (RandomReal[] - 0.5), {i, 1, Nmat}];
deltalist = alist dlist - blist clist;
taulist = alist + dlist ;
disclist = taulist taulist - 4 deltalist ;
fracsaddle = Total[1 - UnitStep[deltalist]] / Length[deltalist] // N;
fracunstablenode = Total[UnitStep[deltalist] UnitStep[taulist] UnitStep[disclist]] / Length[taulist] // N;
   {Nmat, fracsaddle, fracunstablenode}, {Nmat, 100 000, 1 000 000, 100 000}];
```

```
ListLinePlot[data[[All, {1, 3}]]]
ListLinePlot[data[[All, {1, 2}]]]
```