Chapter 2

Displaying and customizing various kinds of plot;

Basic animation

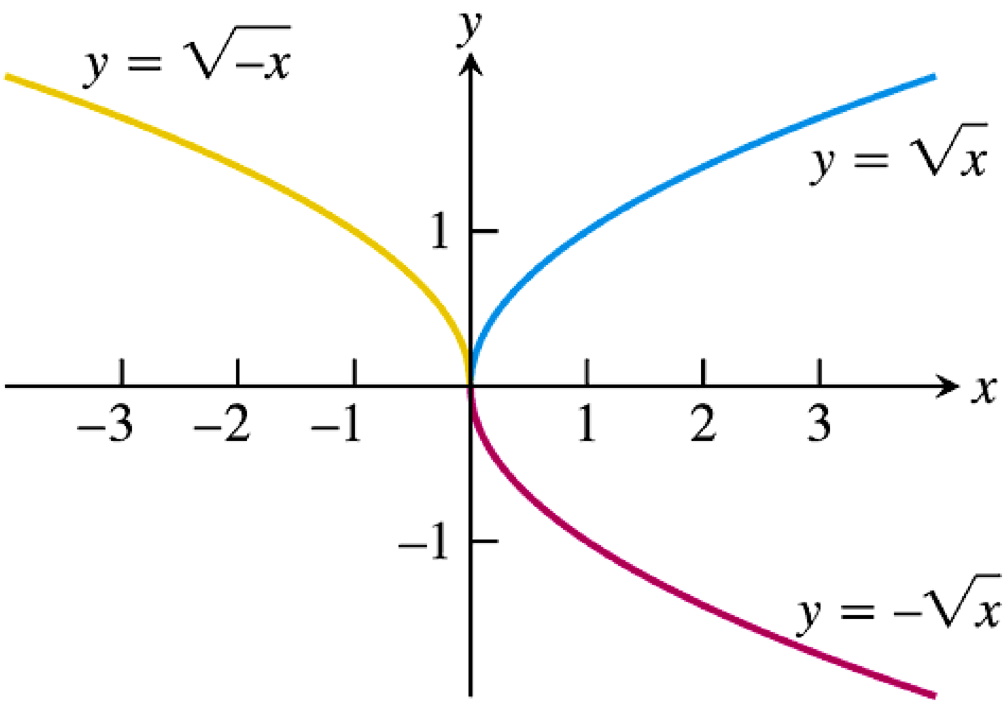


FIGURE 1.59 Reflections of the graph $y = \sqrt{x}$ across the coordinate axes (Example 5c).

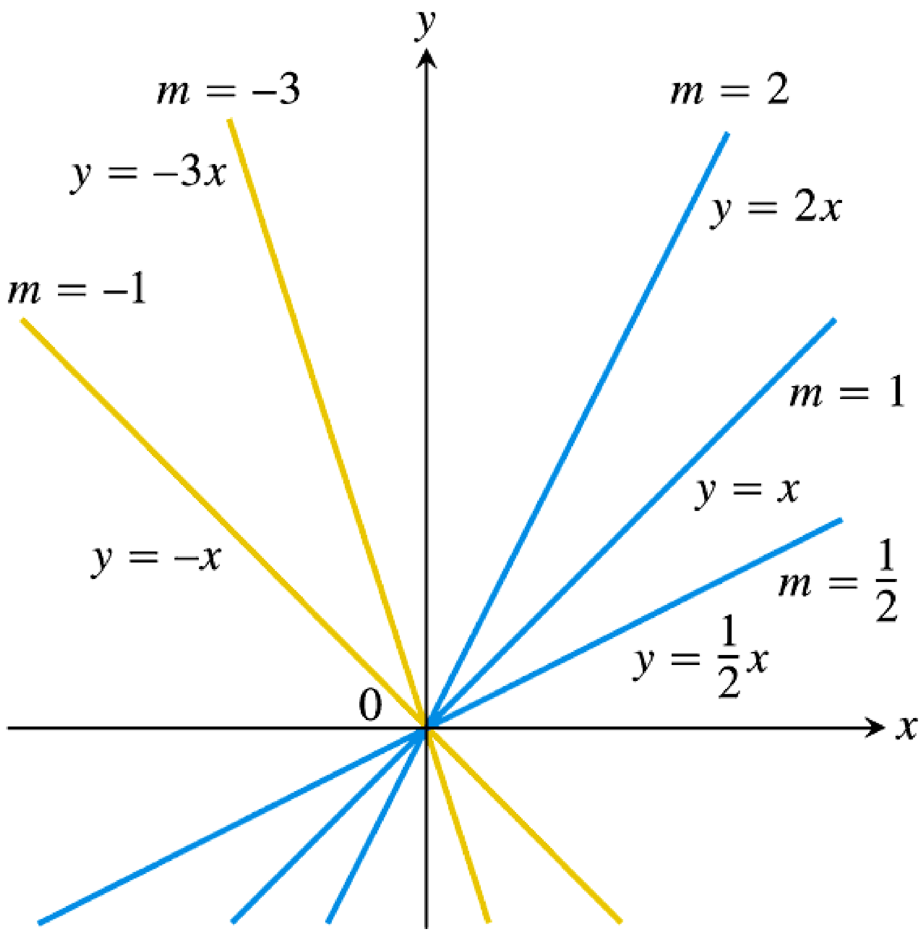


FIGURE 1.34 The collection of lines $y = mx$ has slope m and all lines pass through the origin.

Plot a few functions on the same graph

- Reproduce the previous plots using Mathematica
- Syntax required:
- **f[x_]:=; Plot; List;**
- To customize the plots:
- **PlotRange;PlotStyle;AxesLable;PlotLabel; PlotLegend**

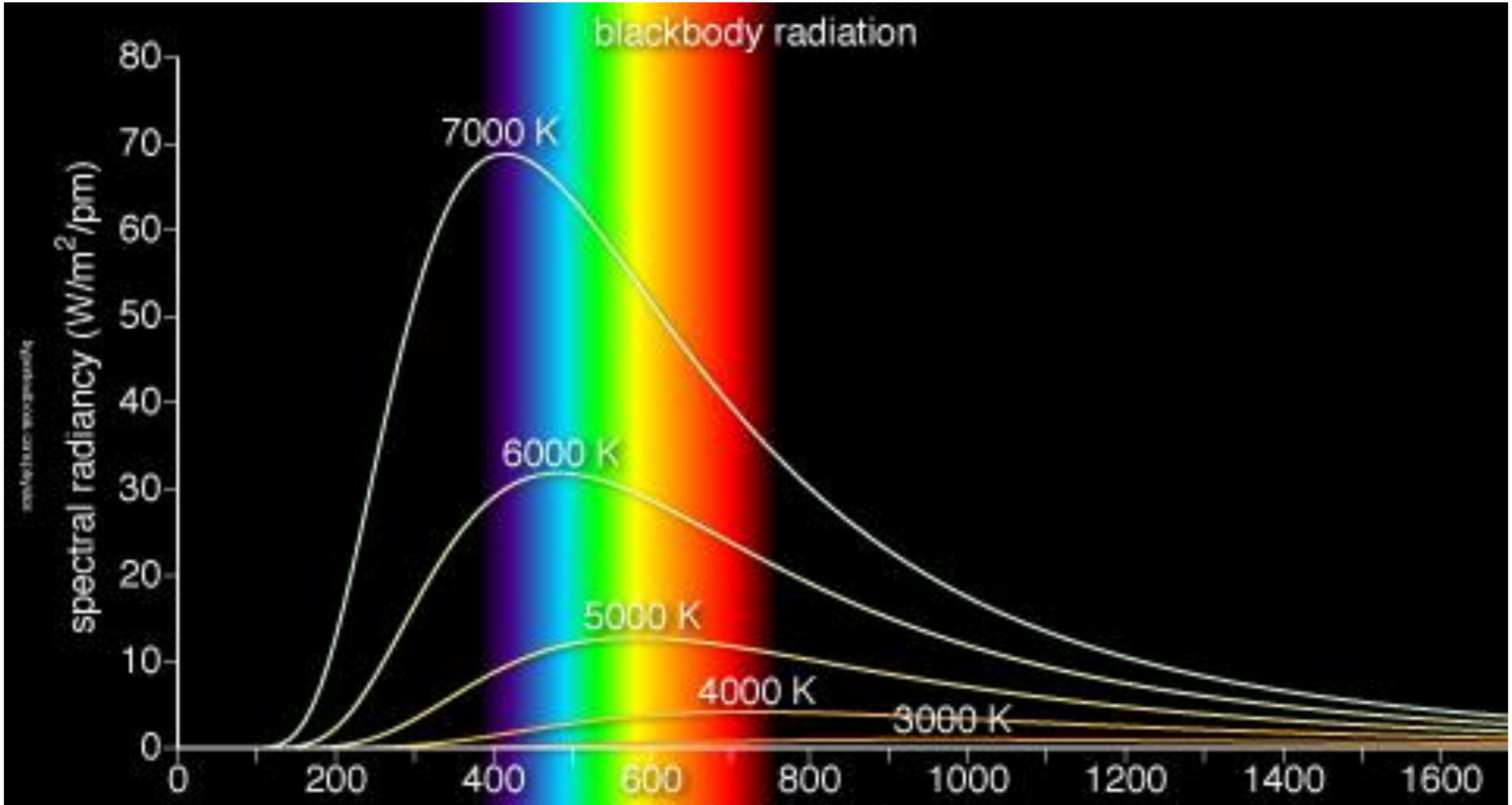
See sample code: C2_plotfunctions.nb

Another example of customizing a function plot

Black Body Radiation: a function of several variables

Black Body Radiation

$$R(\lambda, T) = \frac{2\pi h c^2}{\lambda^5 \left(e^{hc/\lambda k T} - 1\right)}$$



Exercise

- Plot Planck's law of black body radiation for various temperatures on the same graph by defining R as a function of two variables.
- Define function of two variables: $R(\lambda, T) = \dfrac{2\pi h c^2}{\lambda^5 \left(e^{hc/\lambda kT} - 1\right)}$
- *h*, *c*,*T*, are constants

Plotting a sum of terms

Instead of an explicit function (as done previously), we plot a series, which is a ‘function’ comprised of the sum of many terms with specified coefficients.

Generating a series using **Sum[]**

- The function $f(x, N_0) = \sum_{n=1}^{n=N_0} x^n$ can be expressed in Mathematica as

f[x_,N0_]:=Sum[x^n,{n,1,N0}]

- Use these to numerically verify that the infinite series representation of a function converges into the generating function.

EXAMPLE 4 Applying Term-by-Term Differentiation

Find series for $f'(x)$ and $f''(x)$ if

$$f(x) = \frac{1}{1-x} = 1 + x + x^2 + x^3 + x^4 + \cdots + x^n + \cdots$$

$$= \sum_{n=0}^{\infty} x^n, \qquad -1 < x < 1$$

EXAMPLE 6 A Series for $\ln(1+x)$, $-1 < x \le 1$

The series

$$\frac{1}{1+t} = 1 - t + t^2 - t^3 + \cdots$$

converges on the open interval $-1 < t < 1$.

$$\ln(1+x) = \int_0^x \frac{1}{1+t}\,dt = t - \frac{t^2}{2} + \frac{t^3}{3} - \frac{t^4}{4} + \cdots\Big]_0^x$$

$$= x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \cdots, \qquad -1 < x < 1.$$

Mathematica sample codes

$f(x)=1/(1-x)$

$f(x)=1/(1+x)$

Show that the power series representations converge to the generating functions within the radius of convergence.

Example 2 Finding Taylor polynomial for e^x at $x = 0$

$$f(x) = e^x \rightarrow f^{(n)}(x) = e^x$$

$$P_n(x) = \sum_{k=0}^{k=n} \left. \frac{f^{(k)}(x)}{k!} \right|_{x=0} x^k = \frac{e^0}{0!}x^0 + \frac{e^0}{1!}x^1 + \frac{e^0}{2!}x^2 + \frac{e^0}{3!}x^3 + \ldots \frac{e^0}{n!}x^n$$

$$= 1 + x + \frac{x^2}{2} + \frac{x^3}{3!} + \ldots \frac{x^n}{n!}$$ This is the Taylor polynomial of order n for e^x

If the limit $n \rightarrow \infty$ is taken, $P_n(x) \rightarrow$ Taylor series.

The Taylor series for e^x is $1 + x + \frac{x^2}{2} + \frac{x^3}{3!} + \ldots \frac{x^n}{n!} + \ldots = \sum_{n=0}^{\infty} \frac{x^n}{n!}$,

In this special case, the Taylor series for e^x converges to e^x for all x.

Mathematica sample codes

$f(x)=\exp(x)$

Show that the Taylor series representations of e^x at $x = 0$ converge to the generating functions for all values of x

Application to selected physical systems

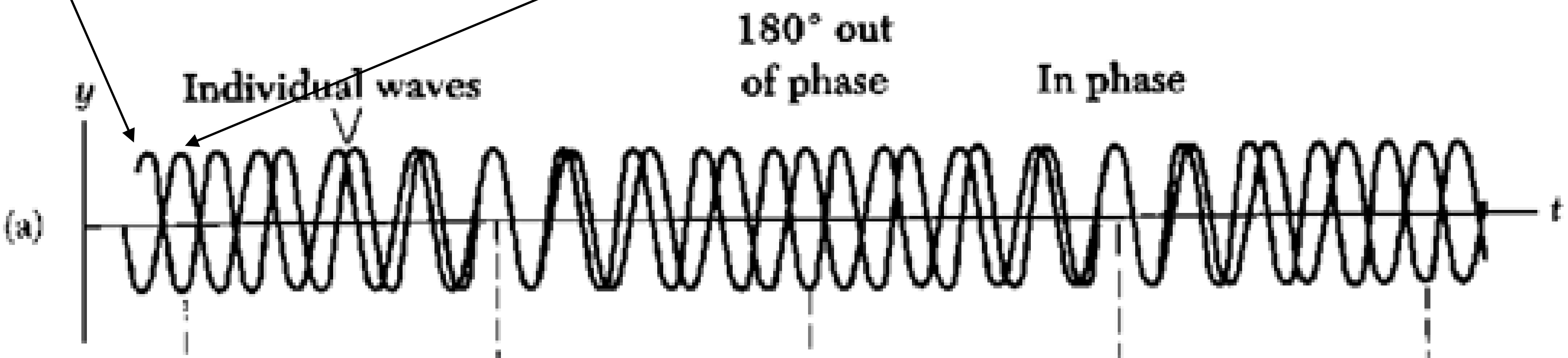
Visualisation of

- wave and wave pulse propagation
- Geometrical optics: Ray-tracing of lens
- 2D projectile motion
- Circular motion
- Elliptic motion
- Simple harmonic motion

Constructing wave pulse

- Two pure waves with slight difference in frequency and wave number $\Delta\omega = \omega_1 - \omega_2$, $\Delta k = k_1 - k_2$, are superimposed

$$y_1 = A\cos(k_1 x - \omega_1 t); \quad y_2 = A\cos(k_2 x - \omega_2 t)$$

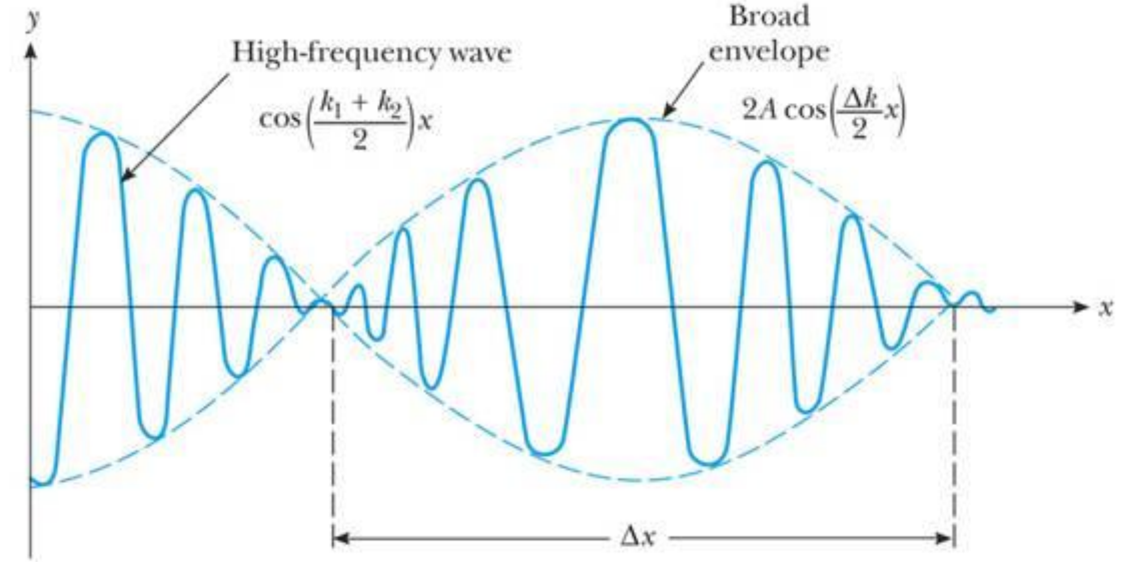


Envelop wave and phase wave

The resultant wave is a ‘wave group’ comprise of an `envelop’ (or the group wave) and a phase waves

$$y = y_1 + y_2$$

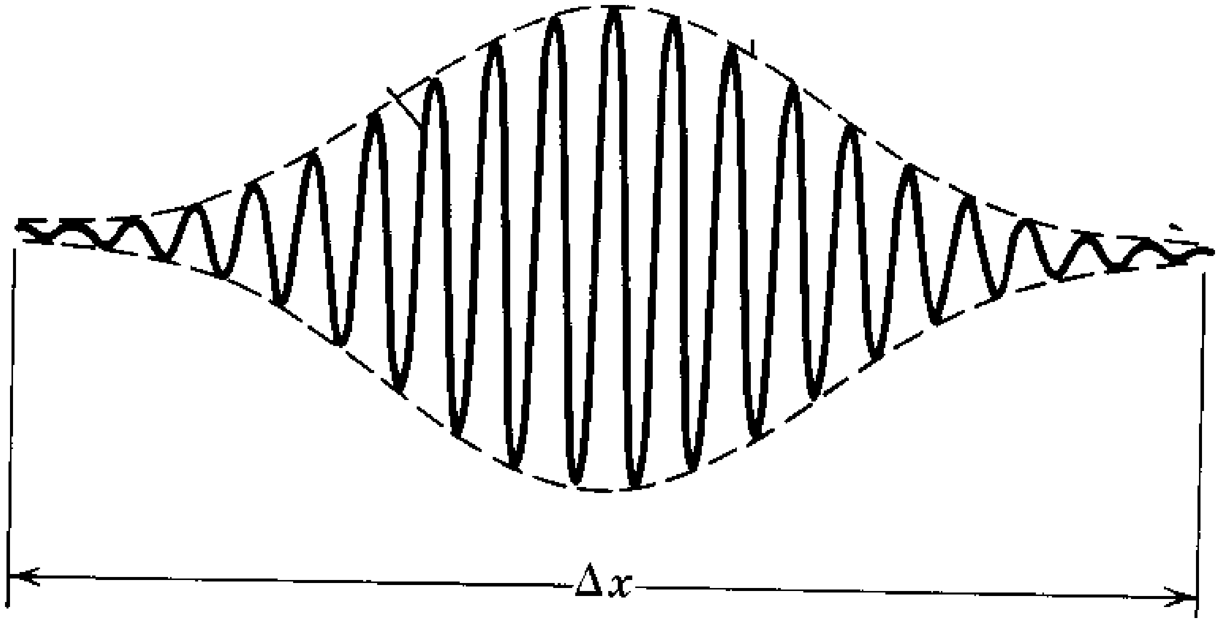
$$= 2A\cos\frac{1}{2}\left(\{k_2 - k_1\}x - \{\omega_2 - \omega_1\}t\right)\cdot\cos\left\{\left(\frac{k_2 + k_1}{2}\right)x - \left(\frac{\omega_2 + \omega_1}{2}\right)t\right\}$$



Wave pulse – an even more \`localised' wave

- In the previous example, we add up only two slightly different wave to form a train of wave group
- An even more \`localised' group wave – what we call a "*wavepulse*" can be constructed by adding more sine waves of different numbers k_i *and* possibly different amplitudes so that they interfere constructively over a small region Δx and outside this region they interfere destructively so that the resultant field approach zero
- Mathematically,

$$y_{\text{wave pulse}} = \sum_{i}^{\infty} A_i \cos(k_i x - \omega_i t)$$



A wavepulse – the wave is well localised within Δx. This is done by adding a lot of waves with with their wave parameters $\{A_i, k_i, \omega_i\}$ slightly differ from each other (i = 1, 2, 3….as many as it can)

Exercise: Simulating wave group and wave pulse

- Construct a code to add *n* waves, each with an angular frequency omegai and wave number ki into a wave pulse for a fixed t.
- Display the wave pulse for t=t0, t=t1, …, t=tn.
- Syntax: **Manipulate**
- Sample code: C2_wavepulse.nb

Ray-tracing of concave and convex lens

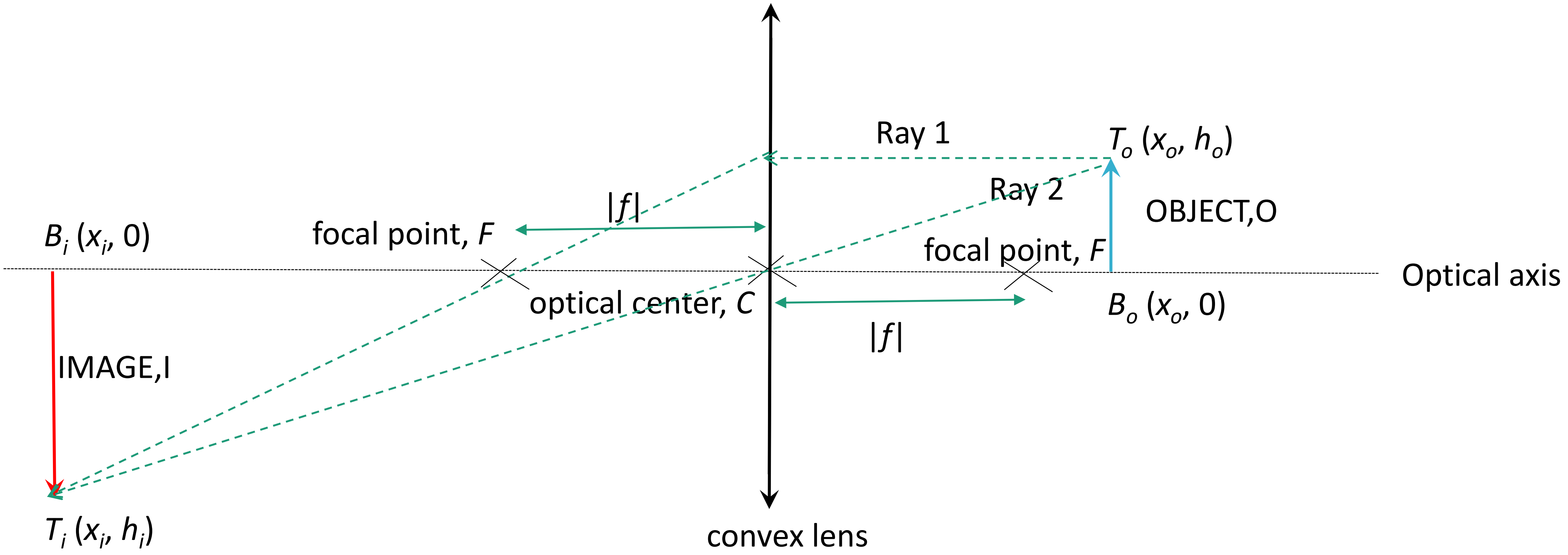
Using **Graphics[Points, Lines]** to

calculate and visualize the image formed by an object in a concave or convex lens

Image formation by a convex lens

- An object with a size h_o placed a distance x_o from the center *C* of a convex lens (with focal length $f > 0$) will form an image with size h_i at a distance x_i from *C*.
- The image can be magnified/diminished, virtual/real or inverted/erect.

Formation of a real, inverted and magnified image in a convex lens by an erected object



Inverted and erected image

- Assume the object is erect, with its tip located at a point T_o (x_o, h_o) above the optical axis, and its base B_o (x_o, 0) located on the optical axis.
- The image in a convex lens can be inverted or erect.
- The image is said to be inverted if its tip T_i (x_i, h_i) is on the opposite side as that of the object's tip, with the base of the object B_i (x_i, 0) located on the optical axis.

Real and virtual image

- If the image is on the same side as that of the object, the image is virtual
- Otherwise, it is real.

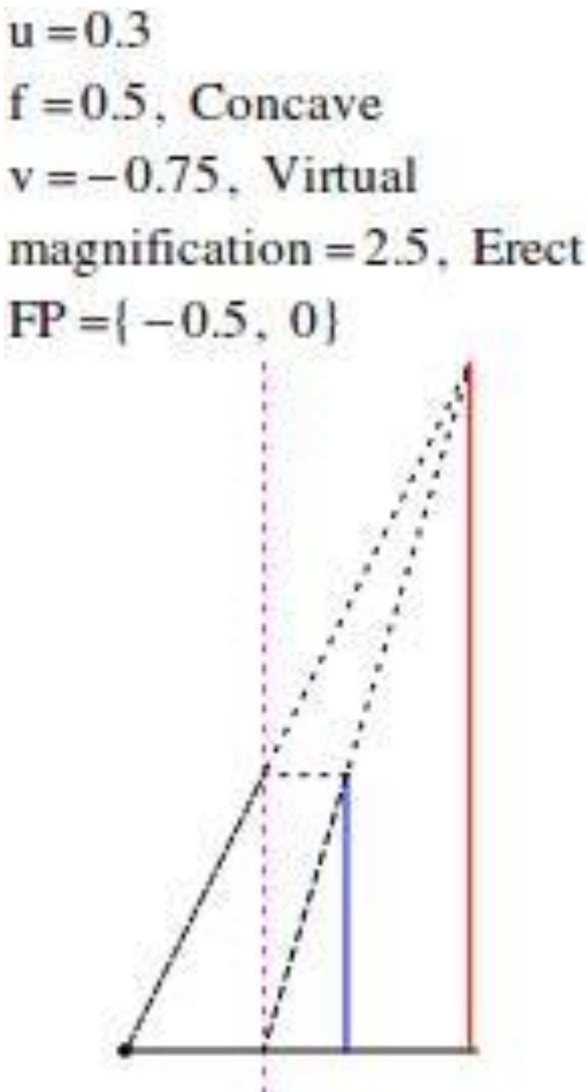
Magnification

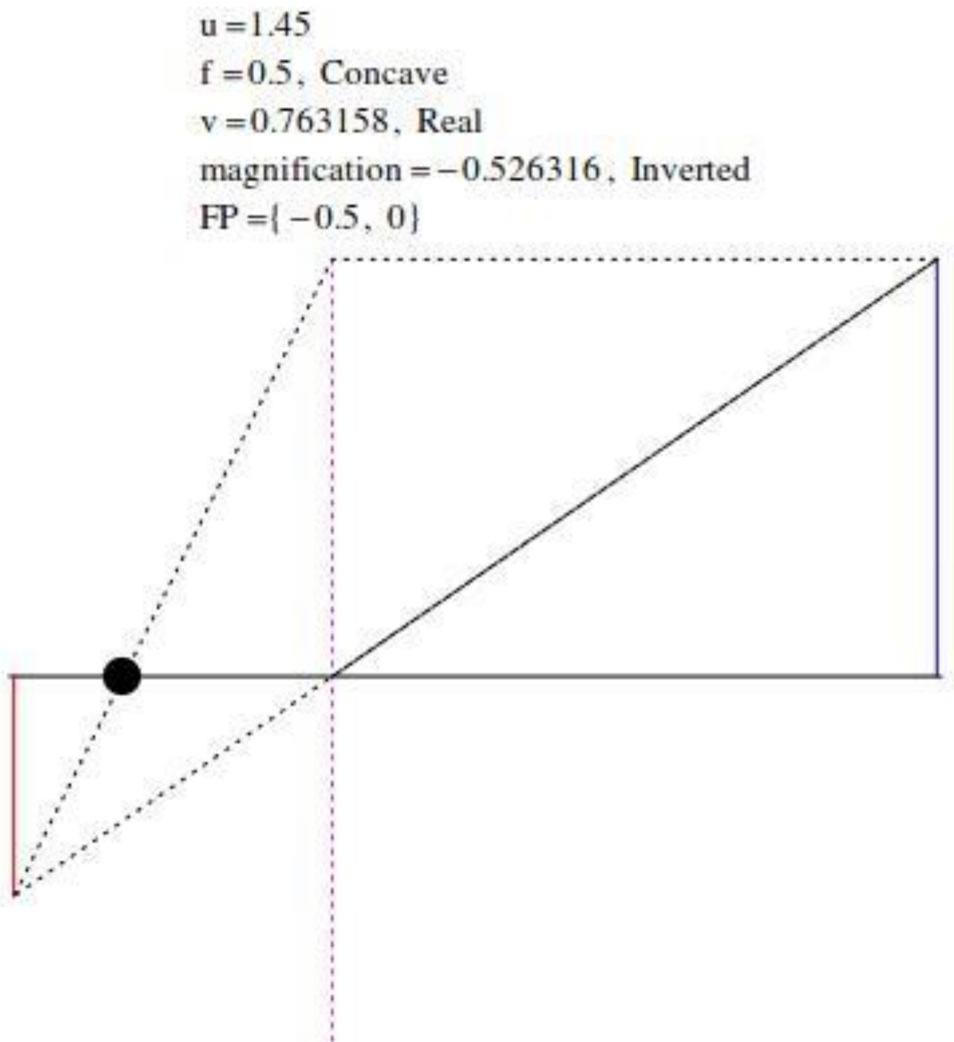
- Magnification of the image is given by $m = \frac{|h_i|}{|h_o|}$.
- If $|m| > 1$, image is magnified; $|m| < 1$, image is diminished.

Formation of image in a convex lens via geometrical ray tracing

- A ray from the tip of object parallel to optical axis (Ray 1) shall go through the focal point on the other side of the lens
- A ray from the tip of the object (Ray 2) shall pass through the lens center *C* in a straight line.
- The intersection of both rays is the location of the tip of image.

Examples of image formation by a convex lens





The coordinates of the image tip $T_i(x_i, h_i)$

- The coordinates of the image tip $T_i(x_i, h_i)$ can be obtained by solving the simultaneous equations of Ray1 and Ray2.

- Ray1: $y_1 = \frac{h_0}{f}x + h_0$
- Ray2: $y_2 = \frac{h_0}{x_0}x$
- Solving $y_1 = y_2$,

$$x_i = \frac{x_0 f}{f - x_0}, h_i = \frac{h_0 f}{f - x_0}$$

Coding exercise for convex lens

Develop a code that reads in supplied values of f of a convex lens, h_o, x_o of an object and does the following:

- visualise the set-up, display the object, image, lens, focal point and optical axis graphically.
- form the image of the object via the geometrical ray tracing method.
- Visualise your output for x_o varies from $0.2f$ till $3f$ at an interval of $0.1f$.

Syntax:

- **Graphics**
- **Point[{P1,P2,P3,…}]**
- **PointSize**
- **Lines[{P1,P2,P3,…}]**
- **Color**
- **Dotted**
- **PlotLabel**
- **Column**

Sample code: C2_ray_tracing_of_convex_lens.nb

Parametric equations for circular motion

- The parametric equations for the *x* and *y* coordinates of an object executing circular motion are given by

$$x(t) = h + R\cos(\omega_0 t),\, y(t) = k + R\sin(\omega_0 t)$$

- C(*h,k*) center of the circle; *R* radius; *t* parameters. For a complete circle, *t* varies from *t*=0 to *t=T*, *T* = period of the circular motion, with $\omega_0 = \frac{2\pi}{T}$ the angular frequency.
- Eliminating the parameter *t* from the parametric equations, the ordinary-looking equation for a circle is deduced:

$$\left(\frac{x-h}{R}\right)^2 + \left(\frac{y-k}{R}\right)^2 = 1$$

Visualising a circle via ParametricPlot[]

- The trajectory can be plotted using **ParametricPlot**.
- You can combine few plots using **Show[]** command.
- **See sample code: C2_circular.nb**

2D projectile motion (recall your Mechanics class)

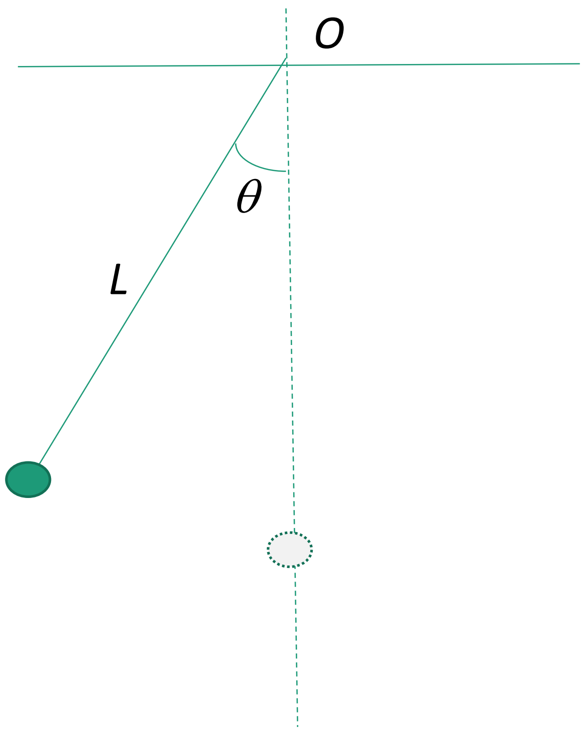
- The trajectory of a 2D projectile with initial location (x_0, y_0), speed v_0 and launching angle θ are given by the equations:
- $x(t) = x_0 + v_0 t \cos\theta; y(t) = y_0 + v_0 t \sin\theta + \frac{g}{2} t^2$, for t from 0 till T, defined as the time of flight, T =-$2(y_0 + v_0 \sin\theta)/g$.
- $g = -9.81;$

2D projectile motion

- Plot the trajectories of a 2D projectile launched with a common initial speed but at different angles
- Plot the trajectories of a 2D projectile launched with a common angle but different initial speed.
- Sample code: C2_2Dprojectile.nb
- For a fixed v0 and theta, how would you determine the maximum height numerically (not using formula)?

Exercise: Simulating SHM

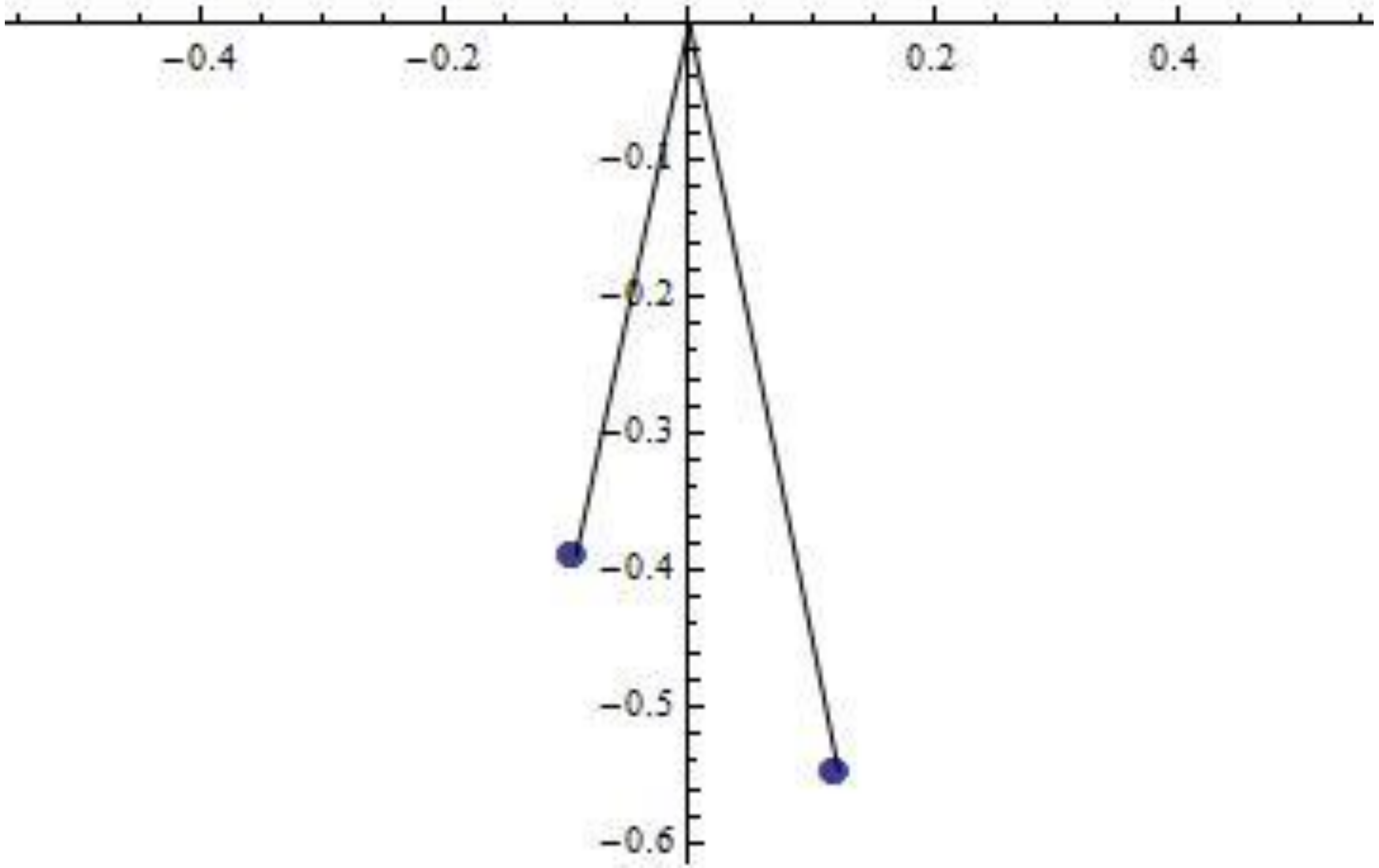
- A pendulum executing simple harmonic motion (SHM) with length *L*, released at rest from initial angular displacement θ_0, is described by the following equations: $\theta(t) = \theta_0 \cos \omega_0 t, \omega_0 = \sqrt{\frac{g}{L}}$. The period *T* of the SHM is given by $T=2\pi/\omega_0$.
- Simulate the SHM using **Manipulate[]**
- Hint: you must think properly how to specify the time-varying positions of the pendulum, i.e., ($x(t)$,$y(t)$).



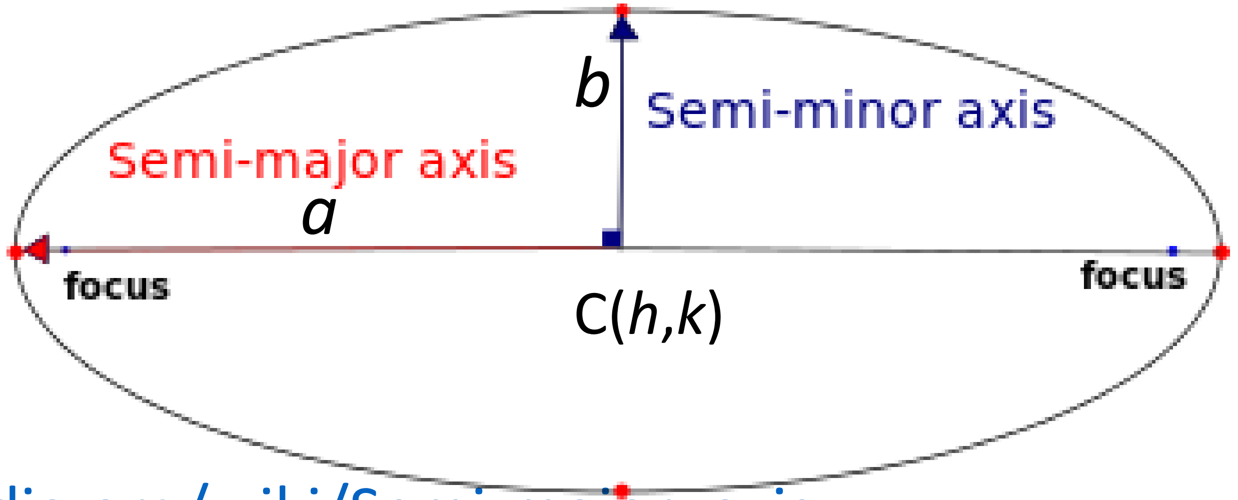
See C2_simulate_pendulum.nb

Exercise: Simulating SHM

- Simulate two SHMs with different lengths L1, L2:
- Plot the phase difference between them as a function of time.

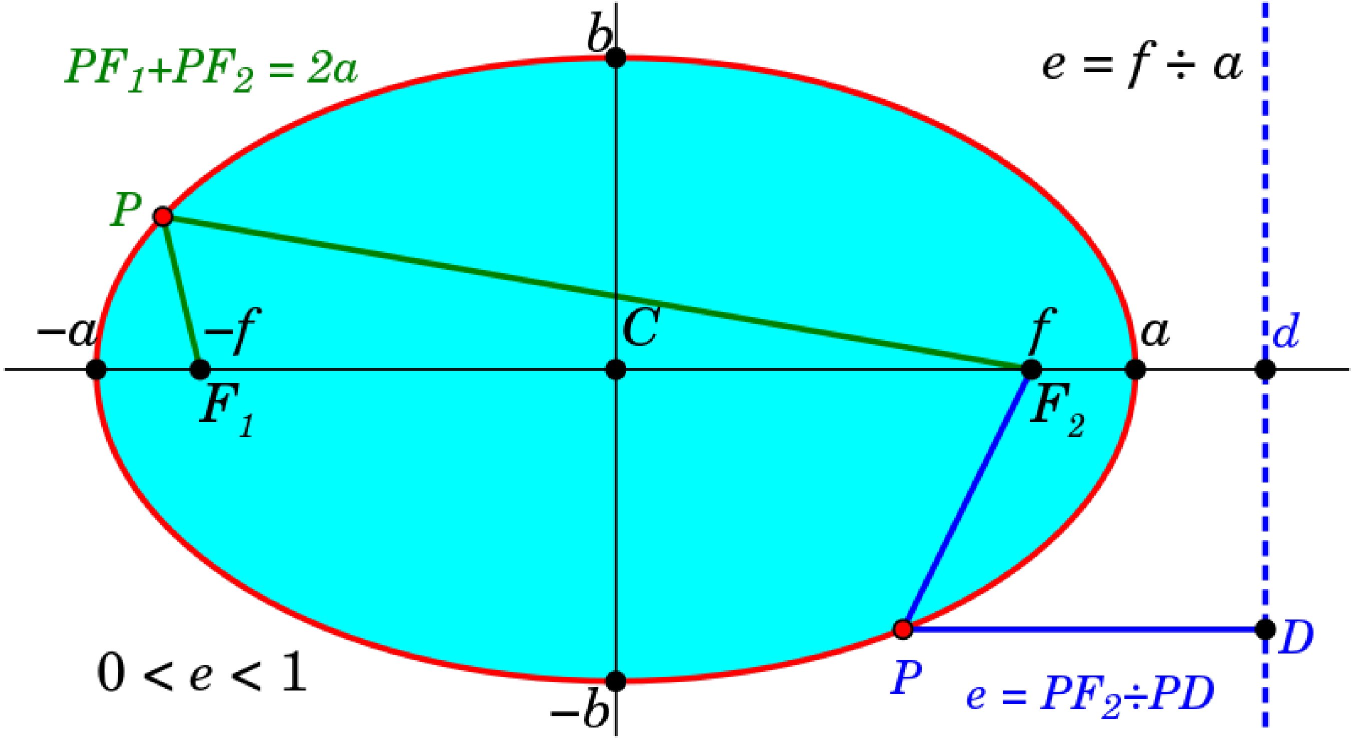


Ellipse



- http://en.wikipedia.org/wiki/Semi-major_axis
- In geometry, the **major axis** of an ellipse is its longest diameter: line segment that runs through the center and both foci, with ends at the widest points of the perimeter. The **semi-major axis, *a*,** is one half of the major axis, and thus runs from the centre, through a focus, and to the perimeter. Essentially, it is the radius of an orbit at the orbit's two most distant points. For the special case of a circle, the semi-major axis is the radius. One can think of the semi-major axis as an ellipse's *long radius*.

Geometry of an ellipse



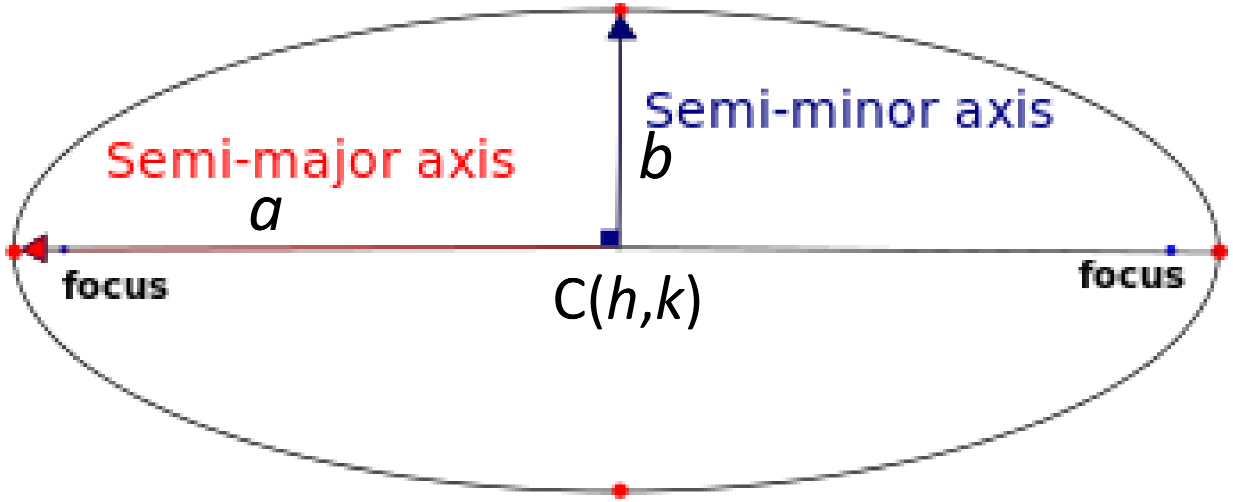
The distance to the focal point from the center of the ellipse is sometimes called the **linear eccentricity**, *f*, of the ellipse.

In terms of semi-major and semi-minor, $f^2 = a^2 - b^2$.

e is the eccentricity of an ellipse is the ratio of the distance between the two foci, to the length of the major axis or $e = 2f/2a = f/a$.

Semimajor and semiminor is related by *e* via $b = \sqrt{1 - e^2}$

Elliptic orbit of a planet around the Sun



- Consider a planet orbiting the Sun which is located at one of the foci of the ellipse.
- The coordinates of the planet at time *t* can be expressed in parametrised form:

$$x(t) = h + a\cos(\omega_0 t),\ y(t) = k + b\sin(\omega_0 t)$$

Elliptic orbit of a planet around the Sun

- Or equivalently,

$$\left(\frac{x-h}{a}\right)^2+\left(\frac{y-k}{b}\right)^2=1$$

where *x*, *y* are the coordinates of any point on the ellipse at time *t, a*, *b* are semi-major and semi-minor.

- C(*h*,*k*) are the coordinates of the ellipse's center.
- ω_0 is the angular speed (a constant) of the planet. ω_0 is related to the period *T* of the planet via T=2π /ω_0
- Note that an ellipse is just a generalization of a circle with its radius now replaced by a semimajor *a* and a semmiinor *b.*
- The period *T* is related to the parameters of the planetary system via $T=2\pi\sqrt{\frac{a^3}{GM}}$, where *M* is the mass of the Sun.

Exercise: Generate an ellipse using **ParametricPlot[]** and **Show[]**

- Display the parametric plot for an ellipse with your choice of *h, k, a, b.*
- Mark also the foci and the center C(*h*,*k*) in your plot.
- See sample code: C2_ParametricPlot_ellipse.nb

- Modify the simple code to also mark the major and minor axes in your plot.
- How would you simulate a point going around the ellipse as time advances?

Simulation of three-body Sun-Planet-Moon

- At this point of time, you should be able to perform a simulation of three-body Sun-Planet-Moon system.

Manipulate List (Array) for measuring a two-body planetary system

- Simulation of the two-body Sun-Earth system using **Grapics[]** and **Points[]** is good for visual display purpose.
- How to perform numerical measurement on the system, e.g., the distance and speed of the planet as a function of time.
- As an illustration, let's measure the distance of the Earth from the Sun, and the speed of the Earth, both as a function of time.
- See sample code: C2_measure_EarthMoon.nb