



MATH & PHYSICS FOR GAME ENGINES

BCA 603 Mathematics and Physics for Game Engines

Serdar Aritan

serdar.aritan@hacettepe.edu.tr

Hacettepe University



MATH & PHYSICS FOR GAME ENGINES

Lecture #2

- Newton's Laws of Motion and Their Applications
- Curve Fitting
- Least Squares Estimation



MATH & PHYSICS FOR GAME ENGINES

Newton's first law, the resultant force acting on a particle must be equal to zero when the particle is at rest or moving with constant velocity in an inertial reference frame:

$$\Sigma \vec{\mathbf{F}} = 0$$

in which $\Sigma \mathbf{F}$ denotes the sum of all forces acting on the particle.

Newton's second law relates the resultant force to the acceleration of the particle

$$\Sigma \vec{\mathbf{F}} = m \vec{\mathbf{a}}$$

in which m is the mass of the particle and a is its acceleration with respect to an inertial reference frame.

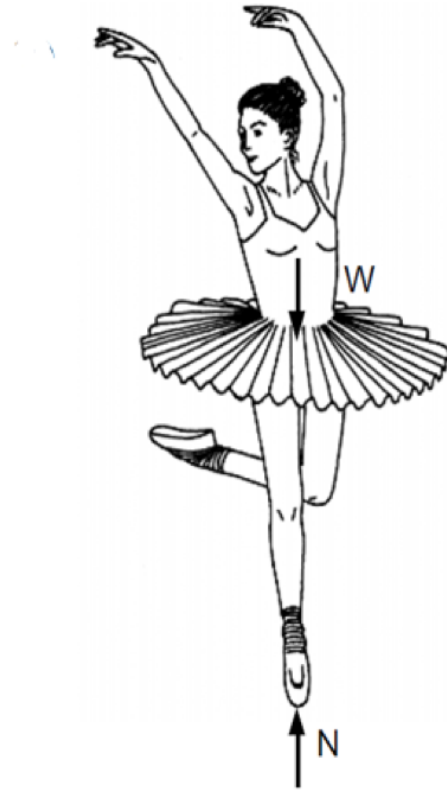
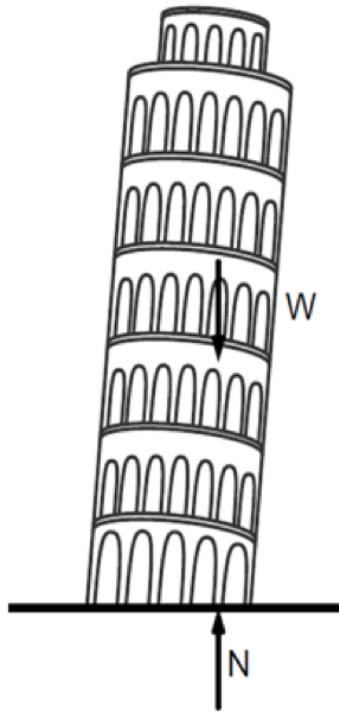
According to the third law, the force of action is equal in magnitude and opposite in direction to the force of reaction

$$\vec{\mathbf{F}}_{1-2} = - \vec{\mathbf{F}}_{2-1}$$

in which $\vec{\mathbf{F}}_{1-2}$ represents the force exerted on particle 1 by particle 2.

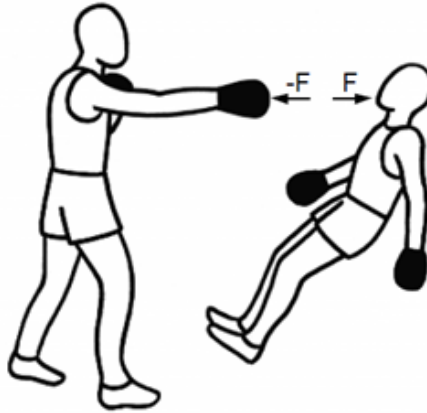


MATH & PHYSICS FOR GAME ENGINES





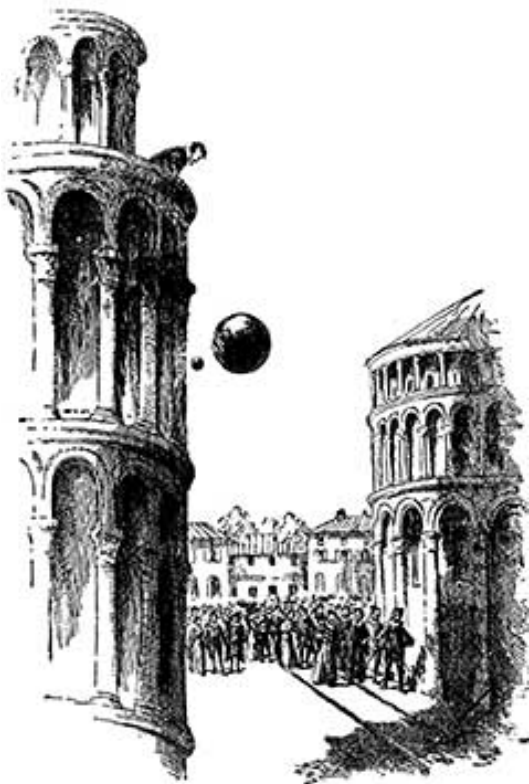
MATH & PHYSICS FOR GAME ENGINES



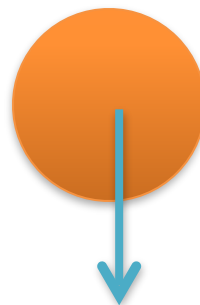


MATH & PHYSICS FOR GAME ENGINES

Free Fall of an Object: An Experiment by Galileo



Galileo didn't know calculus (because Newton and Leibniz hadn't discovered it yet) so he couldn't derive the equation mathematically. Since we do know calculus we know that acceleration is the variation of velocity with time.

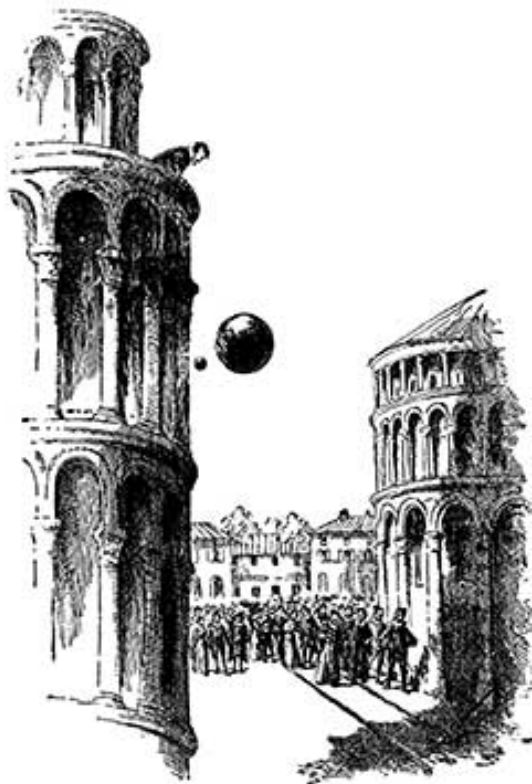


$$\vec{F}_{gravity} = m \vec{g}.$$



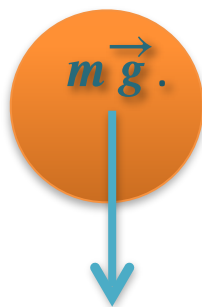
MATH & PHYSICS FOR GAME ENGINES

Free Fall of an Object: An Experiment by Galileo



We assume no drag/air resistance

$$\vec{F}_{gravity} = m \vec{g}$$



$$-\vec{g} = \frac{dv}{dt} = \frac{d^2x}{dt^2}$$

$$dv = -\vec{g} dt$$

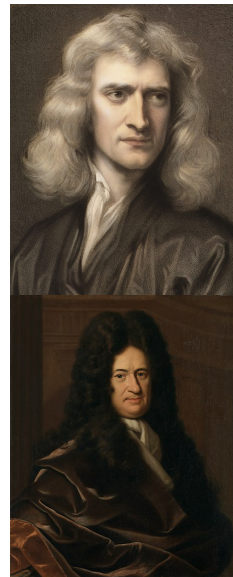
$$v = \int -\vec{g} dt$$

$$v = -\vec{g} t$$

Newton, Leibniz, Euler, Lagrange

$$\dot{f} = \frac{df}{dt} = \partial_t f = f'$$

$$\ddot{f} = \frac{d^2f}{dt^2} = \partial_{tt} f = f''$$

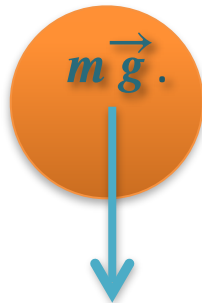
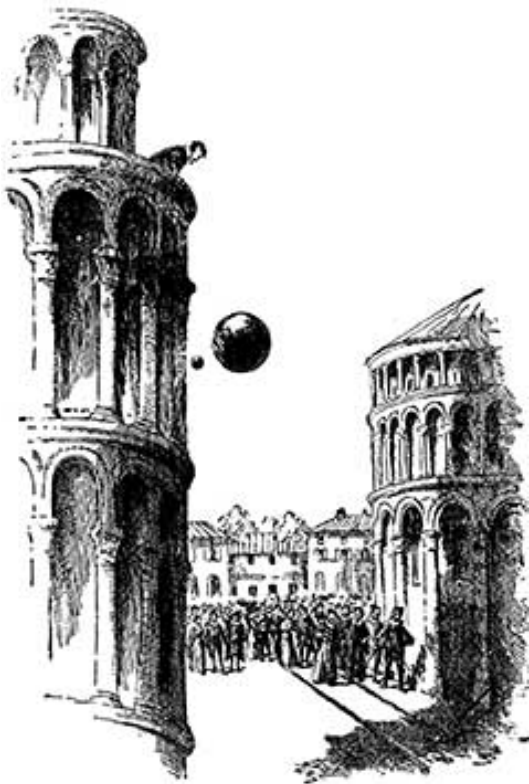




MATH & PHYSICS FOR GAME ENGINES

Free Fall of an Object: An Experiment by Galileo

We assume **no** drag/air resistance



$$\int v dt = \int -\vec{g} t dt$$

$$x = -\frac{1}{2} \vec{g} t^2$$



MATH & PHYSICS FOR GAME ENGINES



SymPy

```
from sympy import *
v, a, t, g, y0 = symbols('v a t g y0', real = True)
init_printing(use_unicode = True)

y = y0 - 0.5*g*t**2
print('Position : ', y)

v = diff(y, t)
print('Velocity : ', v)

a = diff(v, t)
print('Acceleration : ', a)
# Prints
Position :  -0.5*g*t**2 + y0
Velocity :  -1.0*g*t
Acceleration :  -1.0*g
```




MATH & PHYSICS FOR GAME ENGINES

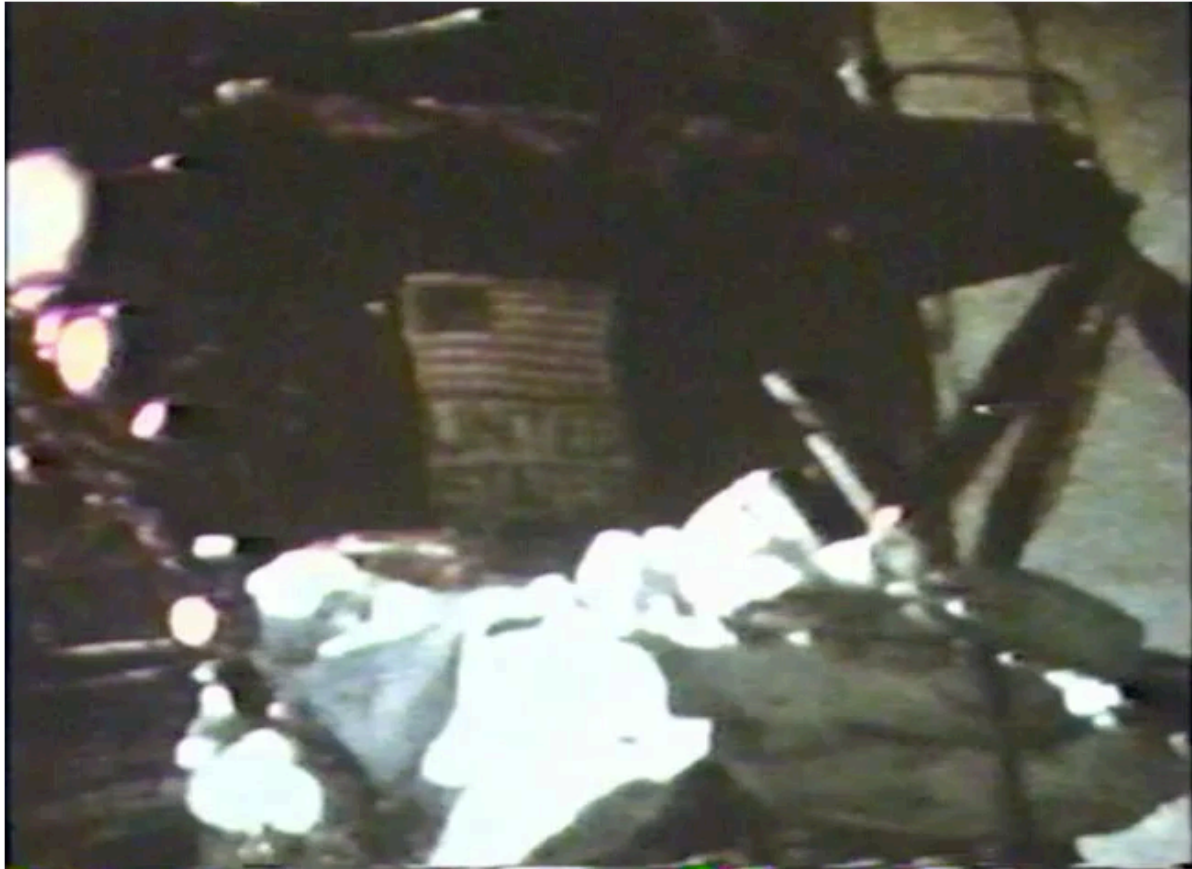
Free Fall of an Object: An Experiment by Apollo15 Team

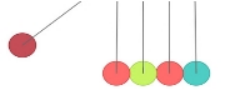


At the end of **the last Apollo 15 moon walk**, Commander David Scott performed a live demonstration for the television cameras. He held out a geologic hammer (1.32 kg) and a feather (0.03 kg) and dropped them at the same time. Because they were essentially in a vacuum, there was no air resistance and the feather fell at the same rate as the hammer, as **Galileo** had concluded hundreds of years before - ***all objects released together fall at the same rate regardless of mass*** of mass.

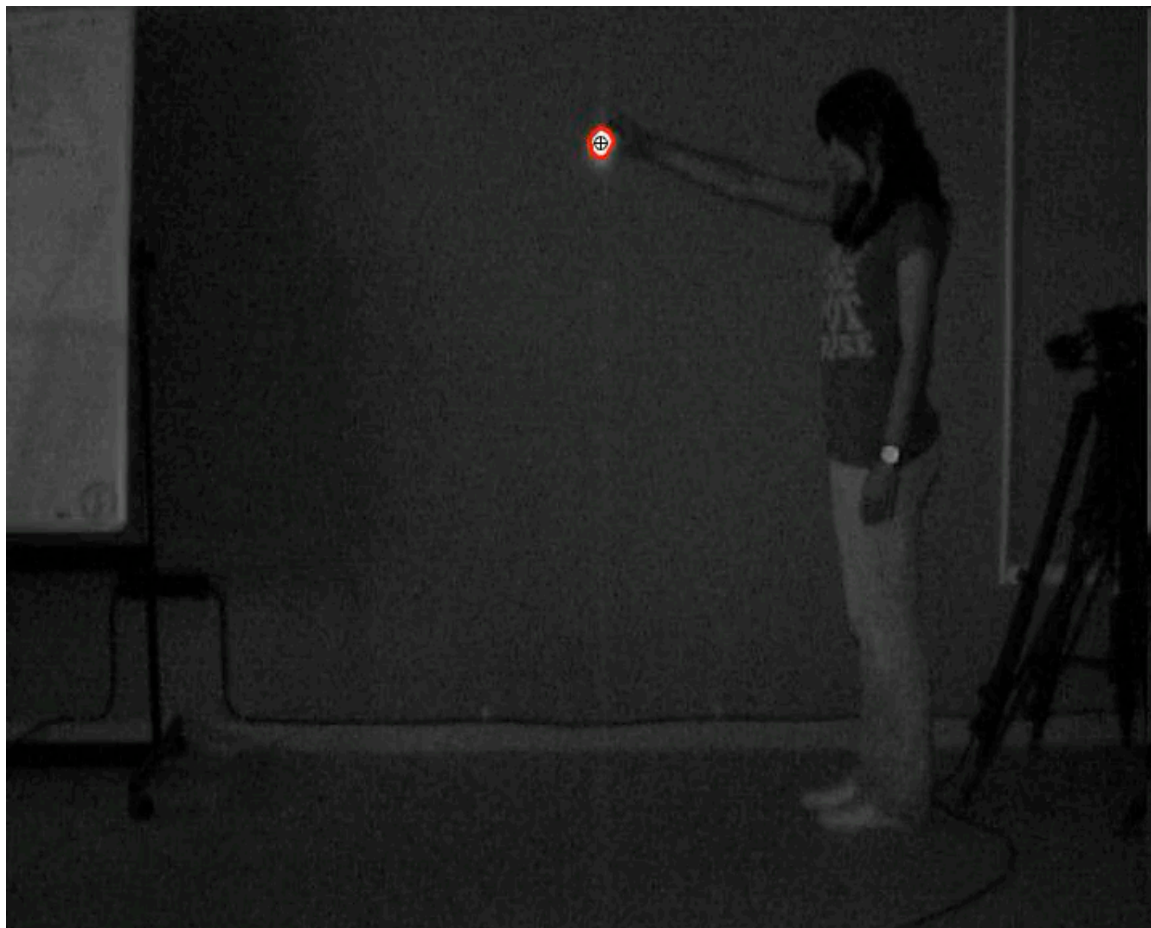


MATH & PHYSICS FOR GAME ENGINES





MATH & PHYSICS FOR GAME ENGINES



X	Y
56.77	159.06
56.74	159.10
56.70	158.85
56.64	158.37
56.53	157.37
56.47	156.11
56.38	154.33
56.30	152.21
56.21	149.65
56.12	146.74
56.03	143.33
55.94	139.58
55.87	135.39
55.80	130.86
55.72	125.80
55.63	120.48
55.56	114.61
55.50	108.47
55.41	101.82
55.38	94.85
55.27	87.41
55.17	79.67
55.07	71.47
55.00	63.06
54.92	54.11
54.79	44.90
54.73	35.26
54.69	25.35
54.61	15.05
54.58	4.53



MATH & PHYSICS FOR GAME ENGINES

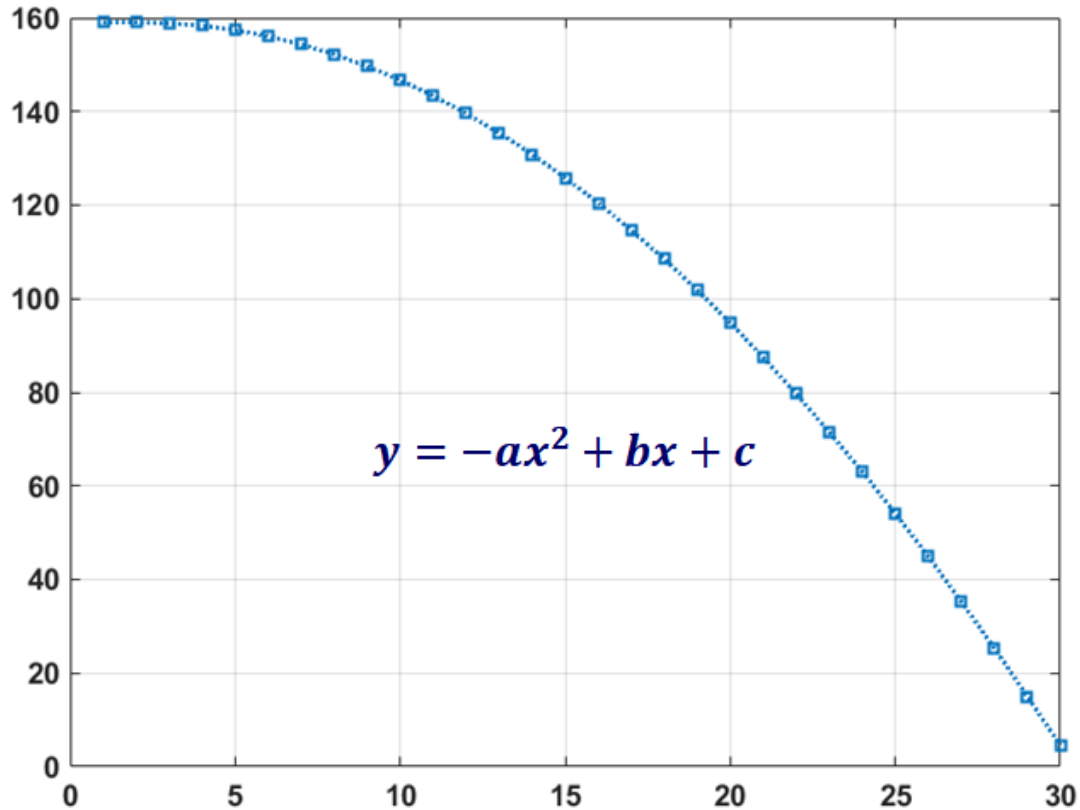
Free Fall of an Object: Table with calculated values

i	t_i (s)	y_i (m)	Δy_i (m)	\bar{v}_i (m/s)	\bar{a}_i (m/s ²)
1	0.0	1.60	-0.05	-0.5	
2	0.1	1.55	-0.15	-1.5	-10.0
3	0.2	1.40	-0.24	-2.4	-9.0
4	0.3	1.16	-0.34	-3.4	-10.0
5	0.4	0.82	-0.43	-4.3	-9.0
6	0.5	0.39			



MATH & PHYSICS FOR GAME ENGINES

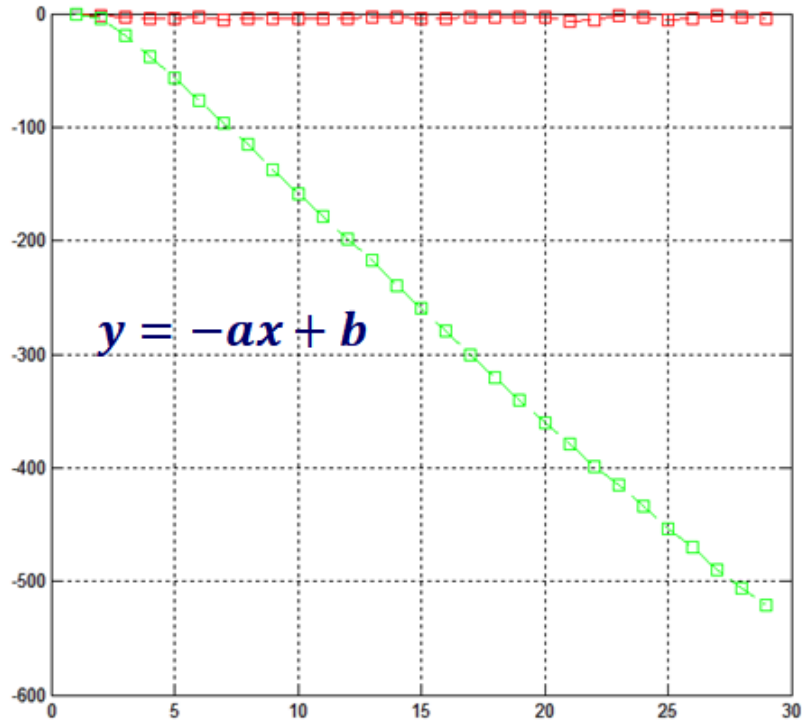
Position in Vertical Direction



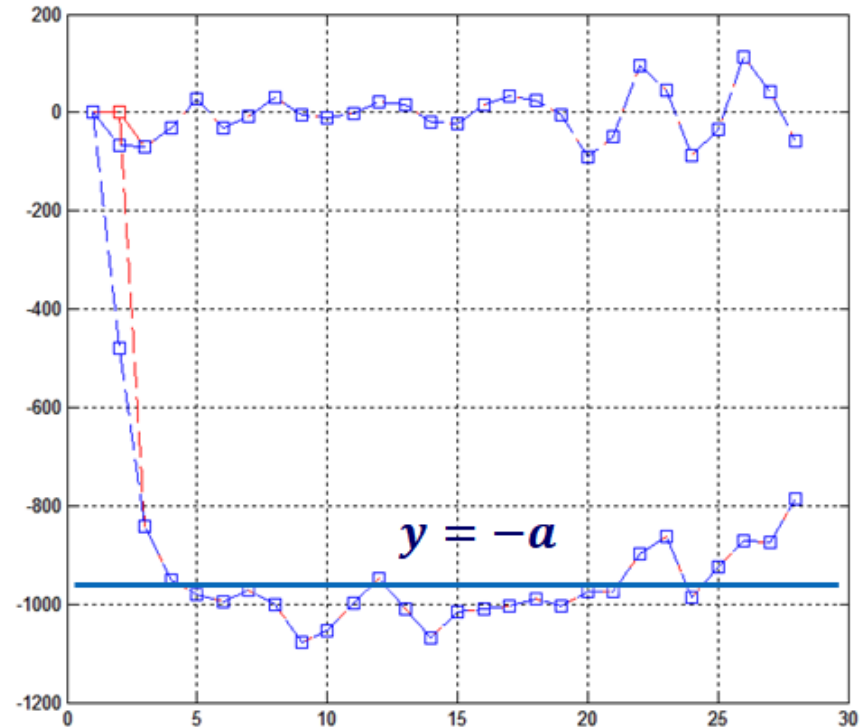


MATH & PHYSICS FOR GAME ENGINES

Velocity



Acceleration





MATH & PHYSICS FOR GAME ENGINES

Curve fitting: Fitting lines and polynomial functions to data points

Curve fitting is the process of constructing a curve, or mathematical function, that has the best fit to a series of data points, possibly subject to constraints. Curve fitting can involve either interpolation, where an exact fit to the data is required, or smoothing.

Most commonly, one fits a function of the form $y=f(x)$.

First degree polynomial equation

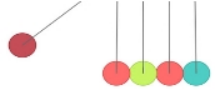
$$y = ax + b$$

Second degree polynomial equation

$$y = ax^2 + bx + c$$

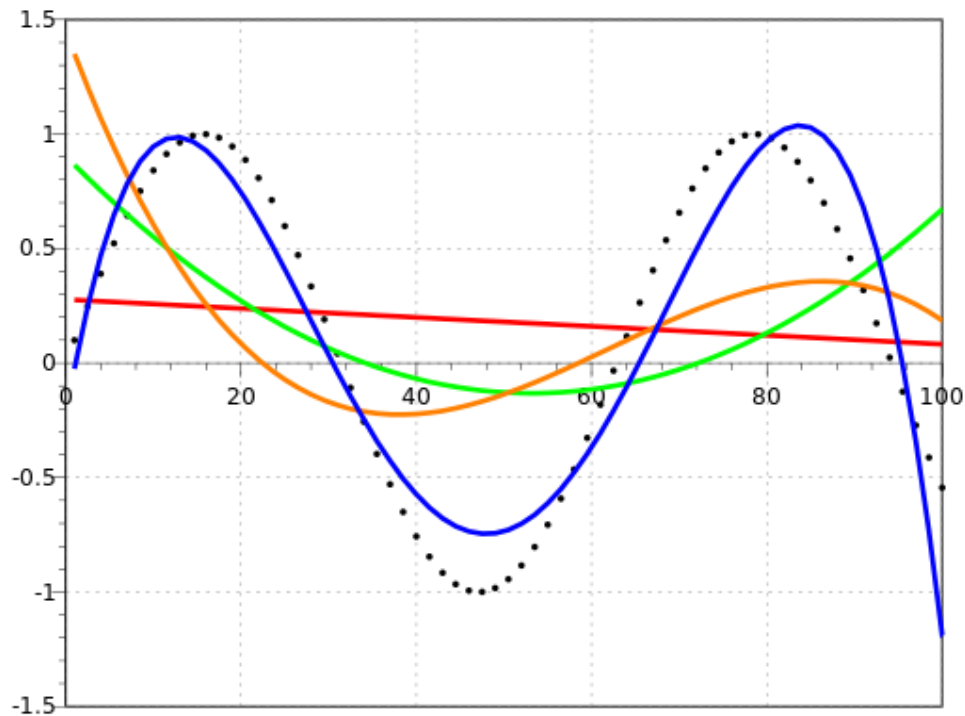
Third degree polynomial equation

$$y = ax^3 + bx^2 + cx + d$$



MATH & PHYSICS FOR GAME ENGINES

Curve fitting: Fitting lines and polynomial functions to data points



Polynomial curves fitting points generated with a sine function.

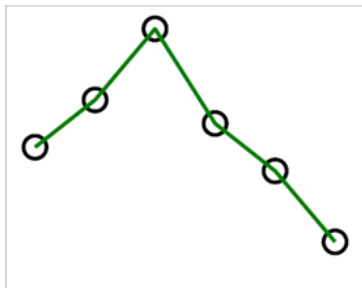
Red line is a first degree polynomial, green line is second degree, orange line is third degree and blue is fourth degree.



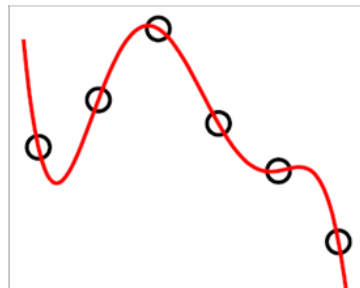
MATH & PHYSICS FOR GAME ENGINES

Curve fitting: Fitting lines and polynomial functions to data points

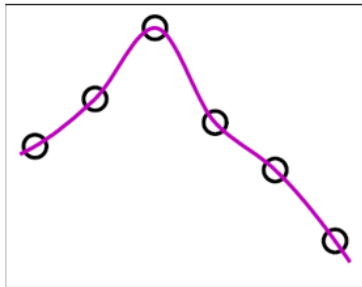
Piecewise linear interpolation



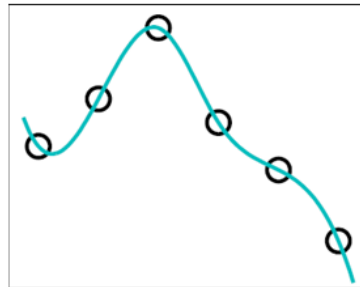
Full degree polynomial interpolation



Shape-preserving Hermite interpolation



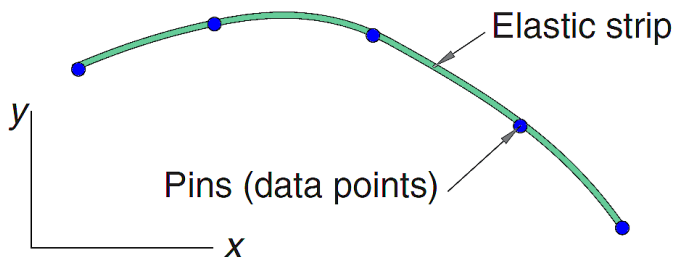
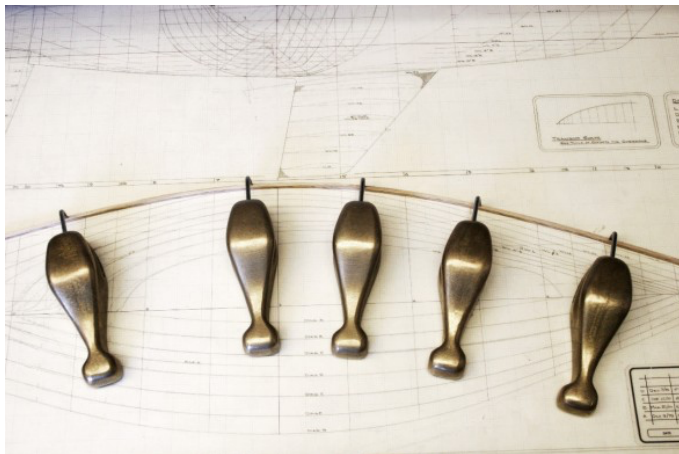
Spline interpolation





MATH & PHYSICS FOR GAME ENGINES

Curve fitting: Fitting lines and polynomial functions to data points

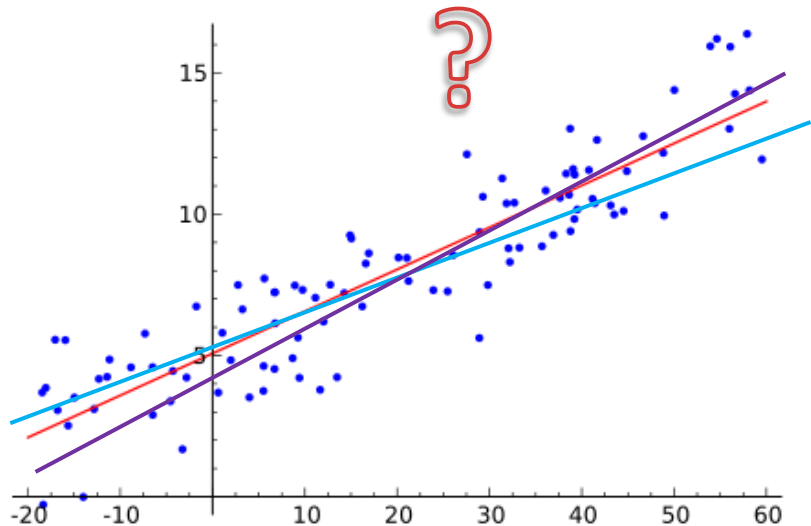


In mathematics a **spline** is a function defined piecewise by polynomials. In interpolating problems, spline is often preferred to polynomial because it yields similar results, even when using low degree polynomials.



MATH & PHYSICS FOR GAME ENGINES

Goodness of fit: Coefficient of determination



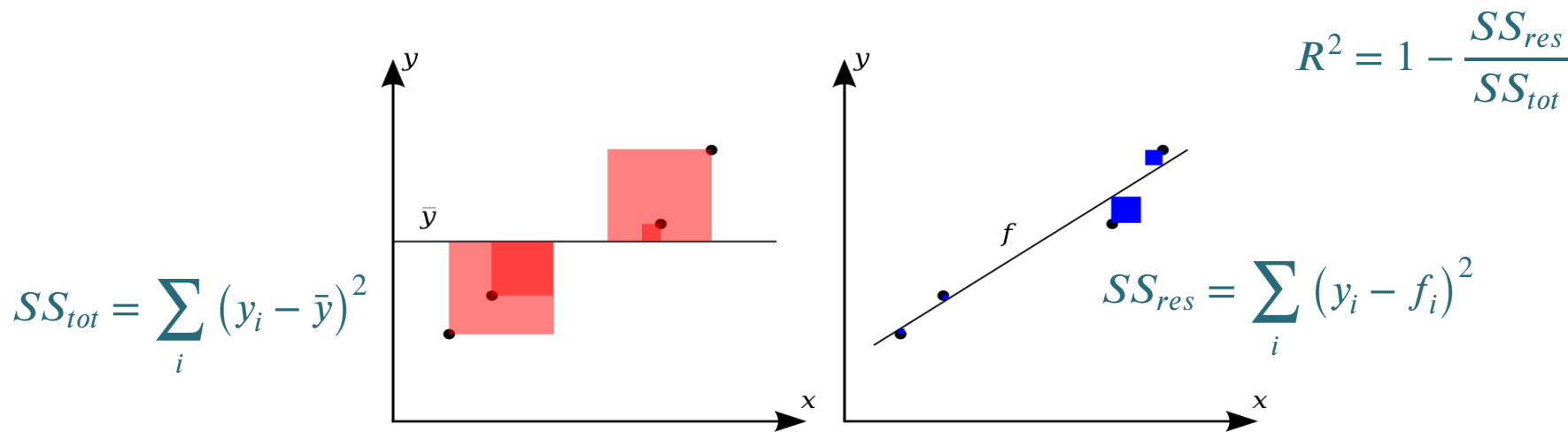
The goodness of fit of a statistical model describes how well it fits a set of observations. Measures of goodness of fit typically summarize the discrepancy between observed values and the values expected under the model in question.

The coefficient of determination, denoted R^2 or r^2 and pronounced ***R squared***, is a number that indicates how well data fit a statistical model.



MATH & PHYSICS FOR GAME ENGINES

Goodness of fit: Coefficient of determination

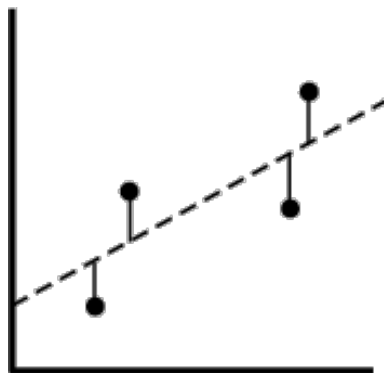


The better the linear regression (on the right) fits the data in comparison to the simple average (on the left), the closer the value of R^2 is to 1. The areas of the blue squares represent the squared residuals with respect to the linear regression. The areas of the red squares represent the squared residuals with respect to the average value.

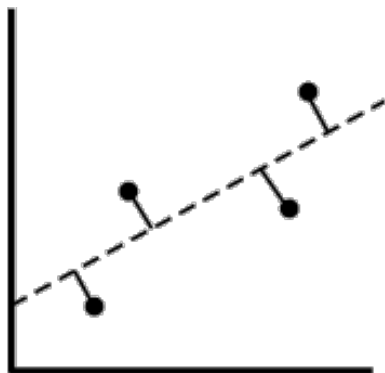


MATH & PHYSICS FOR GAME ENGINES

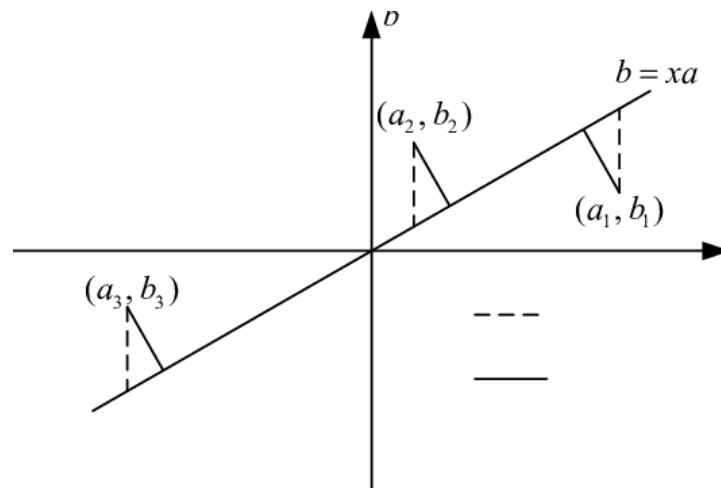
Ordinary Least Squares (OLS) versus Total Least Squares (TLS)



vertical offsets

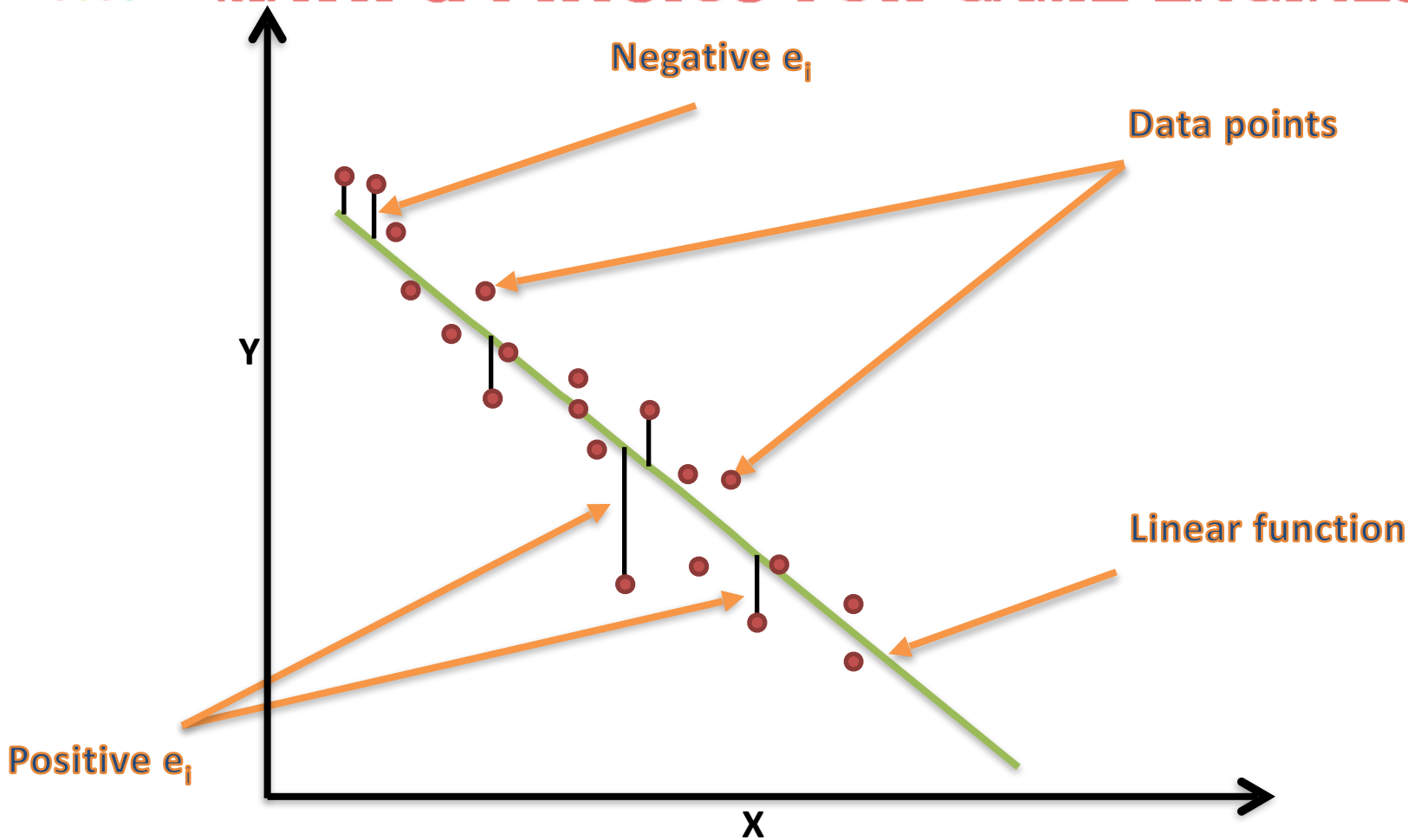


perpendicular offsets





MATH & PHYSICS FOR GAME ENGINES





MATH & PHYSICS FOR GAME ENGINES

Curve fitting: Fitting lines and polynomial functions to data points

How does it work if the function f is to be of the form

$$f(x) = ax + b$$

for to-be-chosen parameters a and b ?

Given (x,y) data pairs $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$

For fixed values of a and b , the mismatch (**error**) is

$$e_1 = ax_1 + b - y_1$$

$$e_2 = ax_2 + b - y_2$$

...

$$e_N = ax_N + b - y_N$$

Goal: make this small

$$\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

by choosing

"data"



MATH & PHYSICS FOR GAME ENGINES



If A is an n -by- m array, and b is an n -by-1 vector, then

```
>>> x = np.linalg.lstsq(A, b)
```

solves the “**least squares**” problem. Namely

- If there is an x which solves $Ax=b$, then this x is computed
- If there is no x which solves $Ax=b$, then an x which minimizes the mismatch between Ax and b is computed.

In the case where many x satisfy one of the criterion above, then a smallest (in terms of vector norm) such x is computed.

So, mismatch is handled first. Among all equally suitable x vectors that minimize the mismatch, choose a smallest one.



MATH & PHYSICS FOR GAME ENGINES

Curve fitting: Fitting lines and polynomial functions to data points

How does it work if the function f is to be of the form

$$f(x) = ax^2 + bx + c$$

for to-be-chosen parameters a , b and c ?

For fixed values of a , b and c , the error at (x_k, y_k) is

$$e_k = ax_k^2 + bx_k + c - y_k \quad \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_N \end{bmatrix} = \begin{bmatrix} x_1^2 & x_1 & 1 \\ x_2^2 & x_2 & 1 \\ \vdots & \vdots & \vdots \\ x_N^2 & x_N & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$



MATH & PHYSICS FOR GAME ENGINES

Curve fitting: Fitting lines and polynomial functions to data points

How does it work if the function f is to be of the form

$$f(x) = a_1x^n + a_2x^{n-1} + \dots + a_nx + a_{n+1}$$

for to-be-chosen parameters a_1, a_2, \dots, a_{n+1} ?

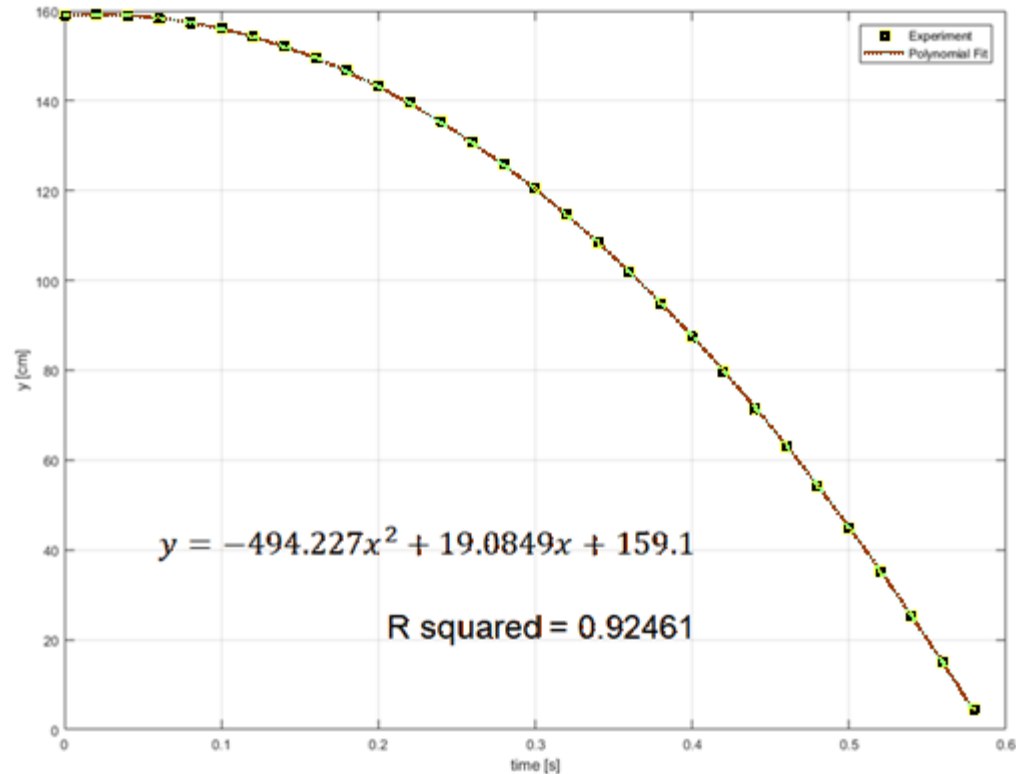
For fixed values of a_1, a_2, \dots, a_{n+1} , the error at (x_k, y_k) is

$$\begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_N \end{bmatrix} = \begin{bmatrix} x_1^n & x_1^{n-1} & \dots & x_1 & 1 \\ x_2^n & x_2^{n-1} & \dots & x_2 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_N^n & x_N^{n-1} & \dots & x_N & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \\ a_{n+1} \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$



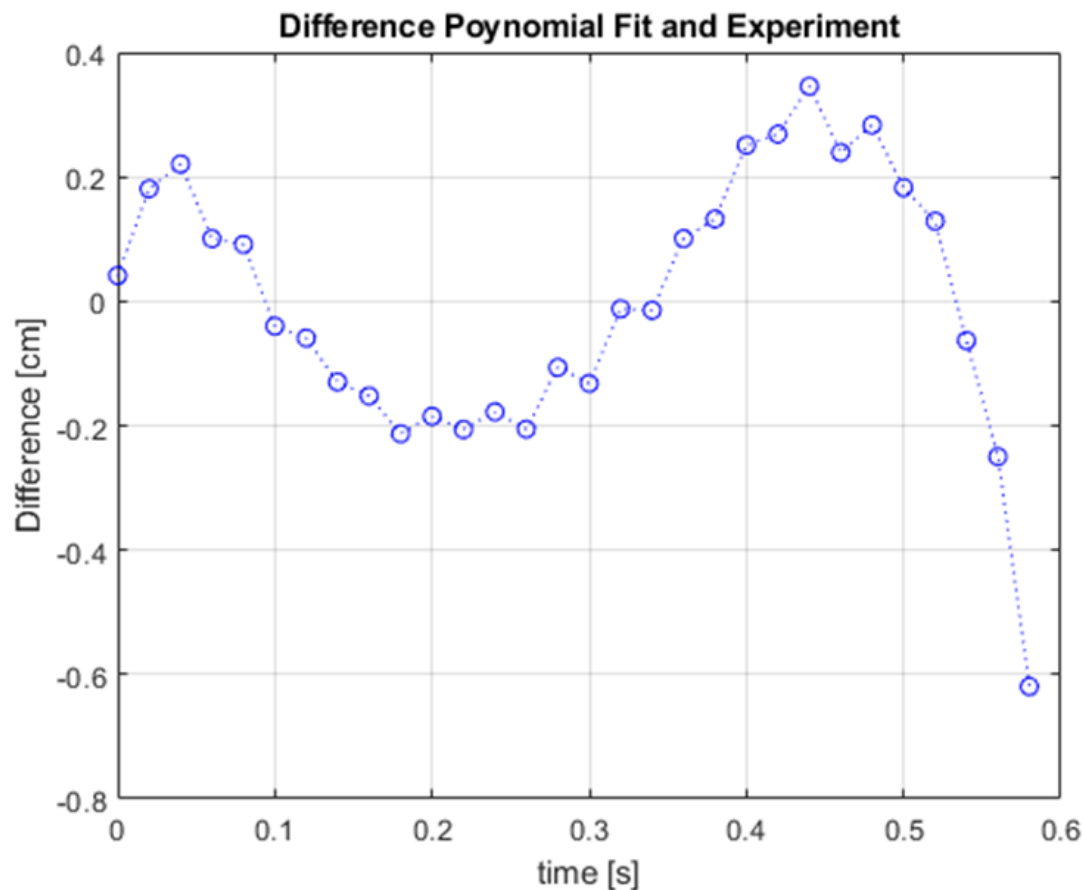
MATH & PHYSICS FOR GAME ENGINES

Curve fitting: 2nd degree polynomial functions to data points





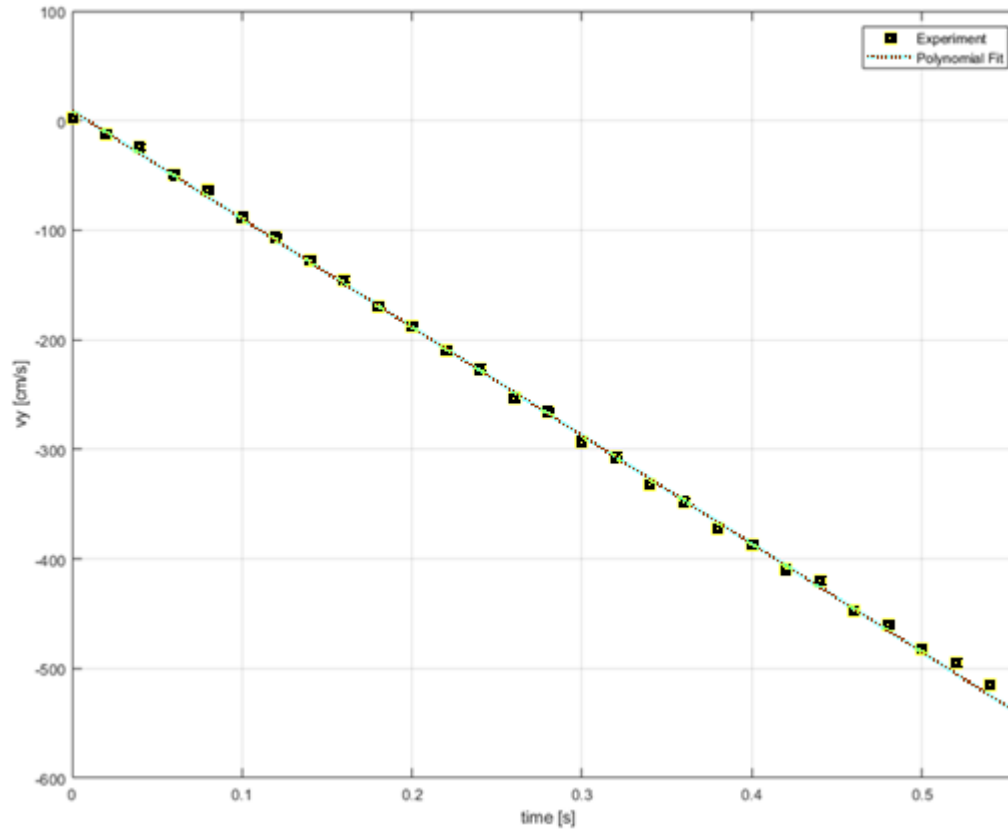
MATH & PHYSICS FOR GAME ENGINES

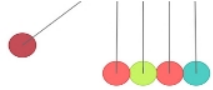




MATH & PHYSICS FOR GAME ENGINES

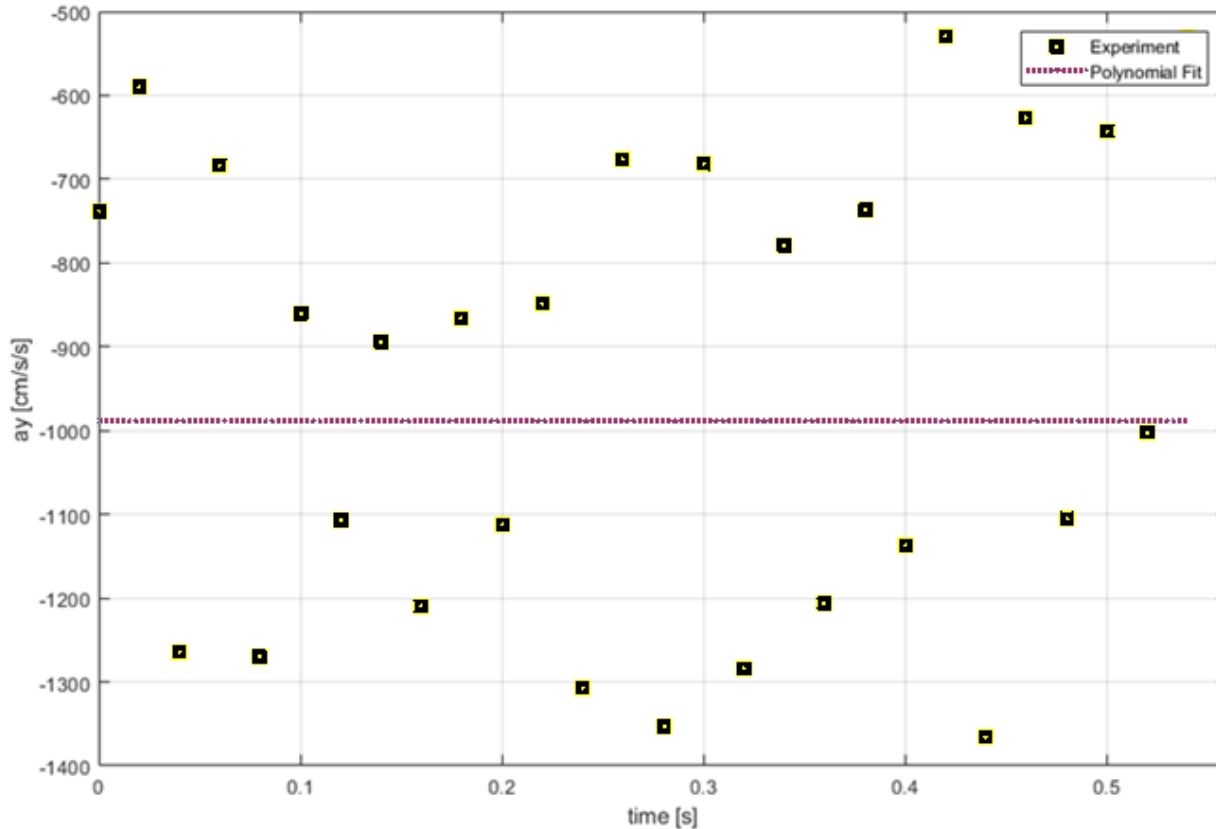
Velocity from the Experiment and Polynomial functions





MATH & PHYSICS FOR GAME ENGINES

Acceleration from the Experiment and Polynomial functions

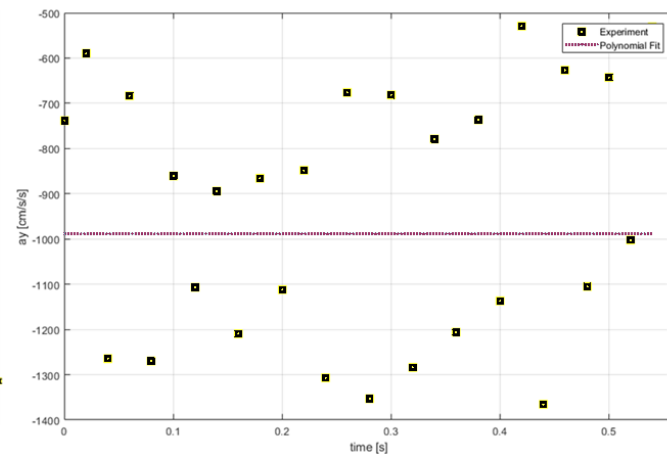
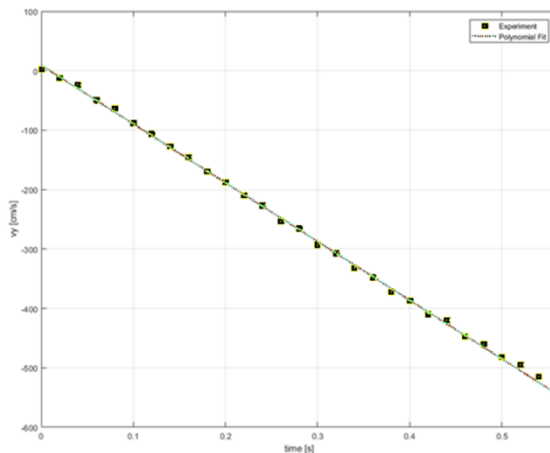
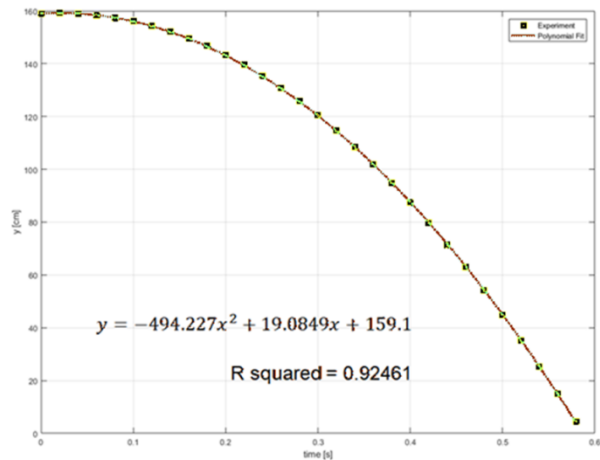


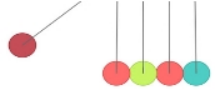


MATH & PHYSICS FOR GAME ENGINES

Calculate the Velocity and Acceleration from the Model

Homework # 2





MATH & PHYSICS FOR GAME ENGINES

Homework #2

Homework Assignment: Due NO LATER than Wednesday 13th March 2025

To: serdar.aritan@hacettepe.edu.tr

Cc: serdar.aritan@gmail.com

Subject : BCA603 HW2