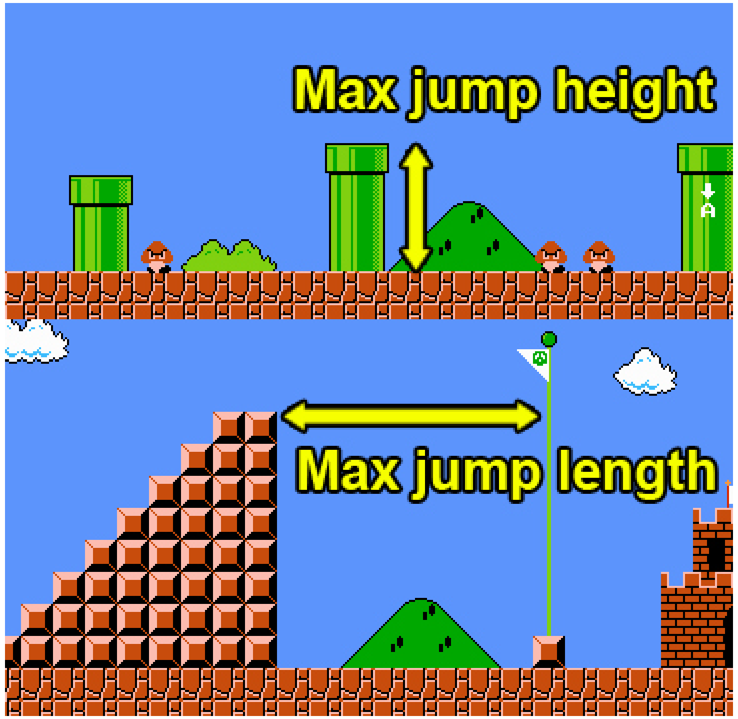
Helge Rhodin

CPSC 427
Video Game Programming

Game Balancing

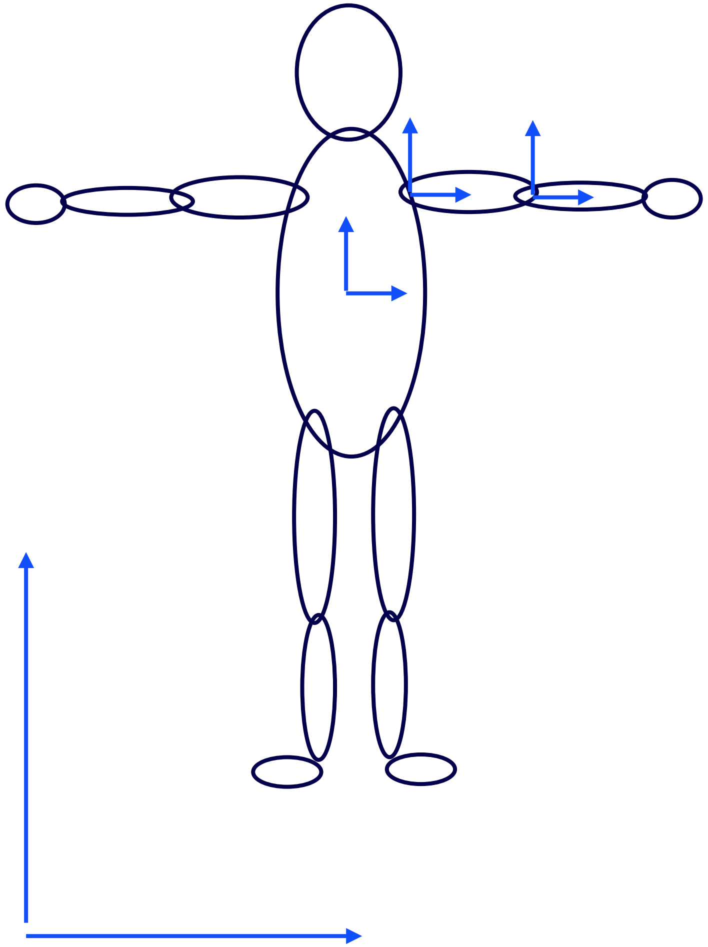


Setup

@Helge: Pressed record?

@Class: Logged into iClicker cloud?

Recap: Transformation Hierarchies



Scenes have multiple coordinate systems

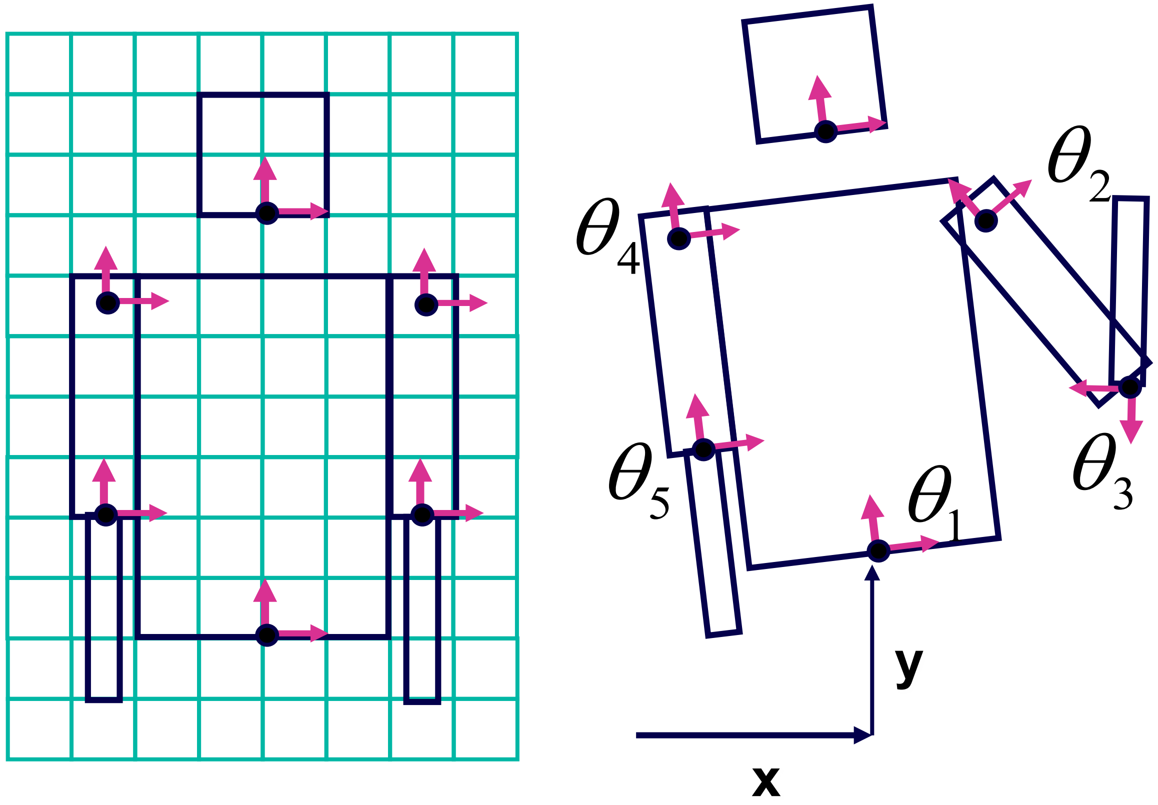
- Often strongly related
  - *Parts of the body*
  - *Object on top of each other*
    - Next to each other…

Independent definition is bug prone

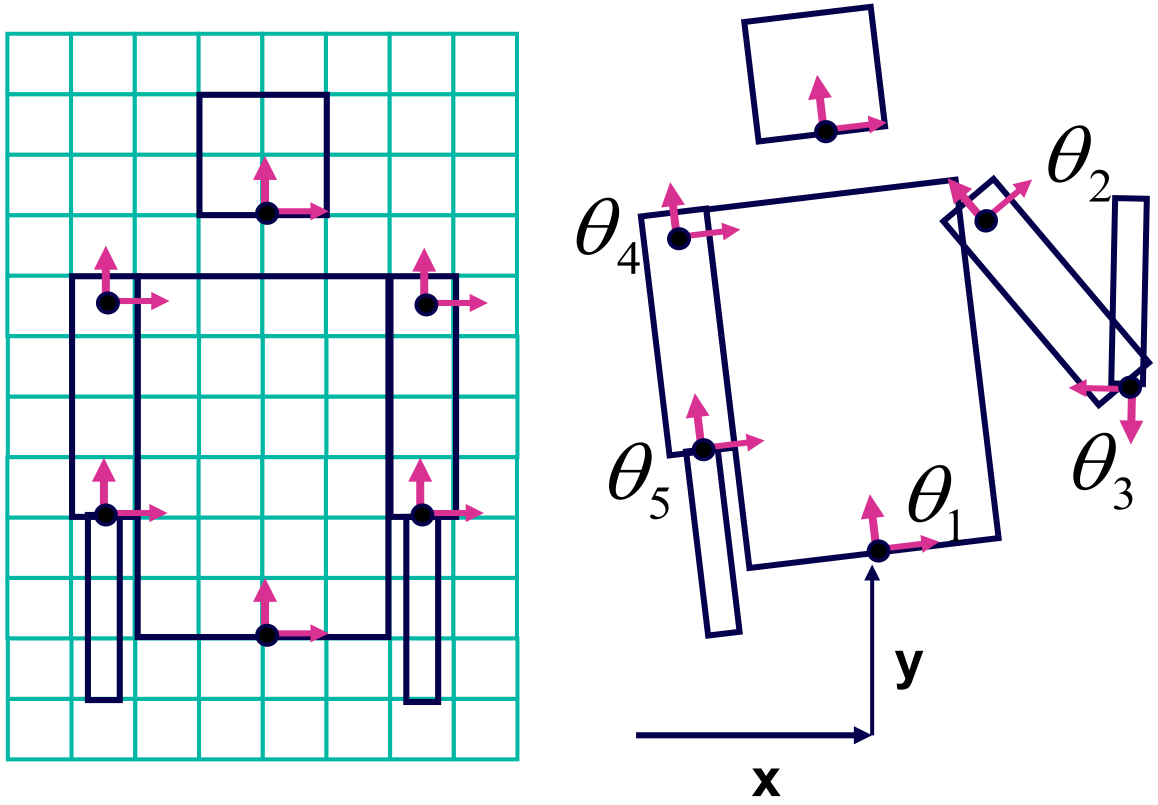
Solution: Transformation Hierarchies

© Alla Sheffer, Helge Rhodin

Recap: Transformation Hierarchy Examples



Recap: Transformation Hierarchy Examples



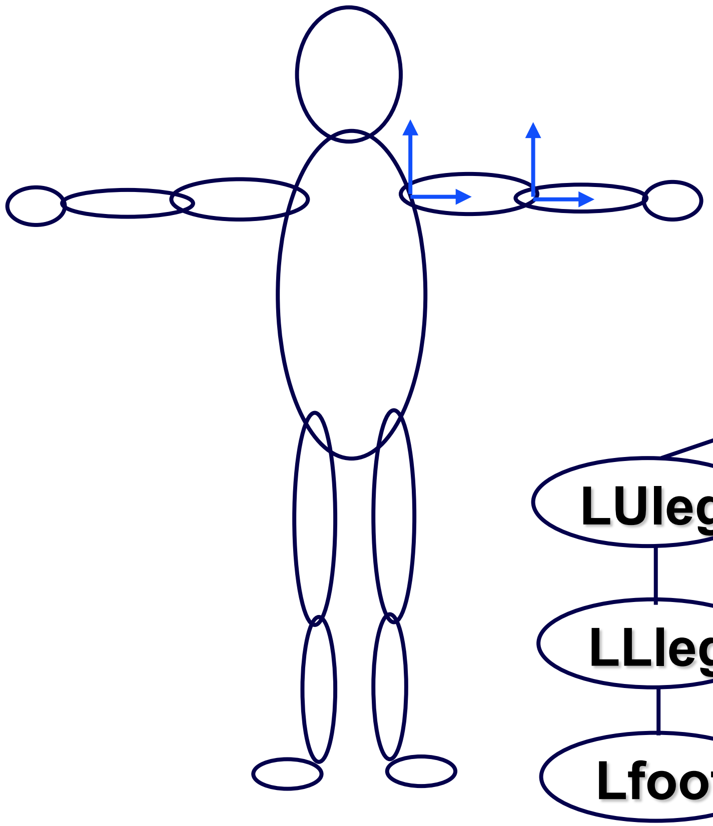
$$M_1 = Tr_{(x,y)} \cdot Rot\theta_1$$

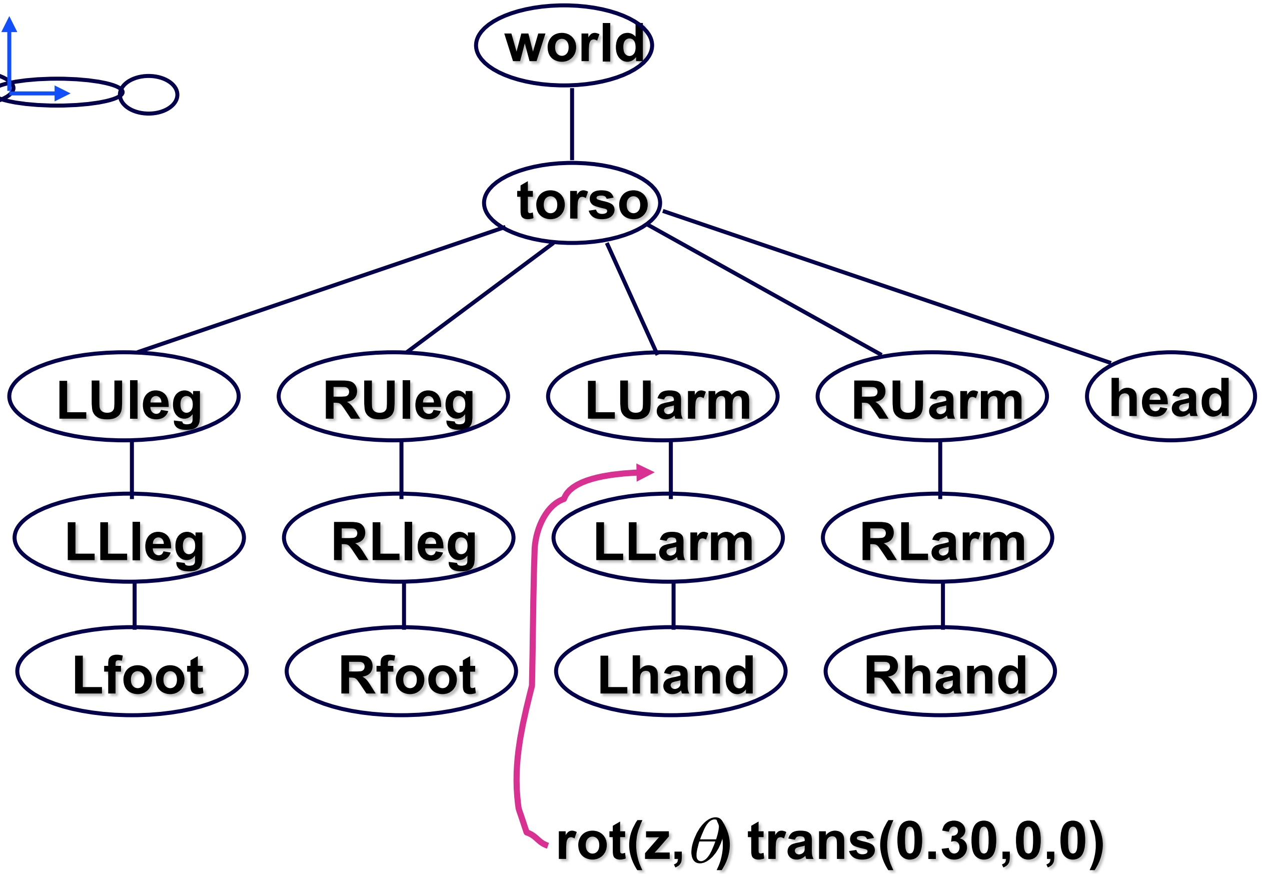
$$M_2 = M_1 \cdot Tr_{(2.5,5.5)} \cdot Rot\ \theta_2$$

$$M_3 = M_2 \cdot Tr_{(0,-3.5)} \cdot Rot\ \theta_3$$

© Alla Sheffer, Helge Rhodin

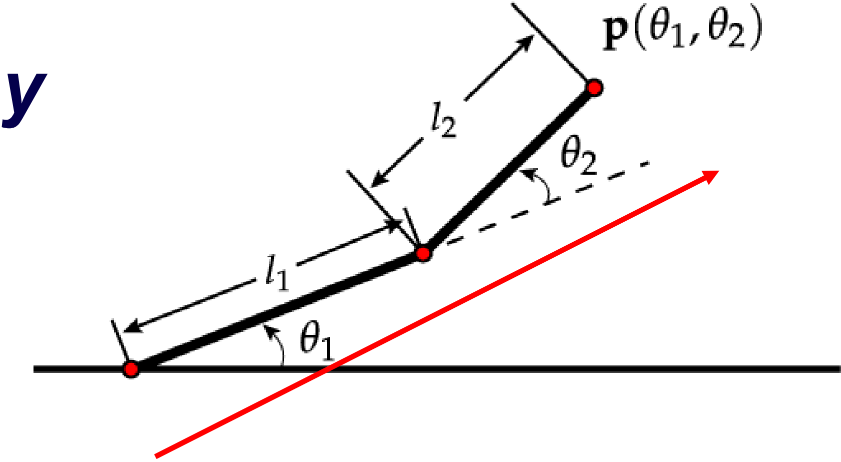
Recap: Transformation Hierarchies





© Alla Sheffer, Helge Rhodin

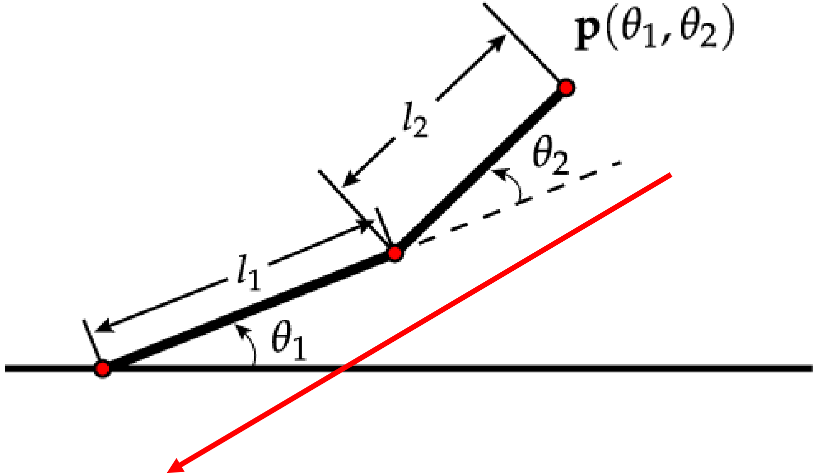
Recap: Forward vs. inverse kinematics



Forward kinematics

- ***given joint axis, angle, and skeleton hierarchy***
- ***compute joint locations***
  - *start at the end-effector (e.g. arm)*
    - rotate all parent joints (up the hierarchy) by θ
  - *iteratively continue from child to parent*

Inverse kinematics



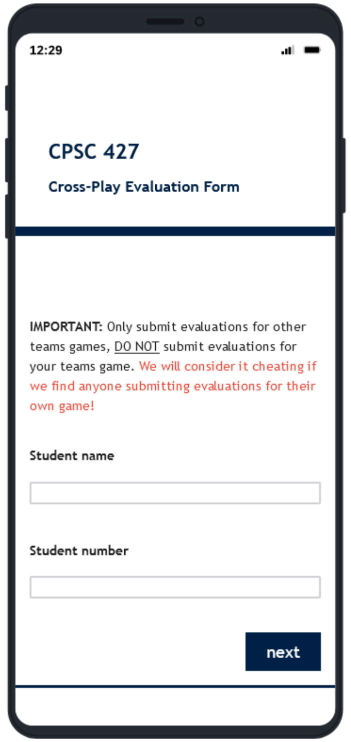
- given skeleton hierarchy and goal location
- optimize joint angles (e.g. gradient descent)
- minimize distance between end effector (computed by forward kinematics) and goal locations

Cross-play

- ***Short lecture until ~3:30 pm***

- ***Cross-play***
  - starting at ~3:30 pm
  - four rounds, each (15 min):
    - *A plays B's game (5-7min)*
      and also B plays A's game (5-7 min)
  - fill out feedback form after each play (5 min)

  - https://piazza.com/class/krpu7s953e6wt?cid=363



12:29

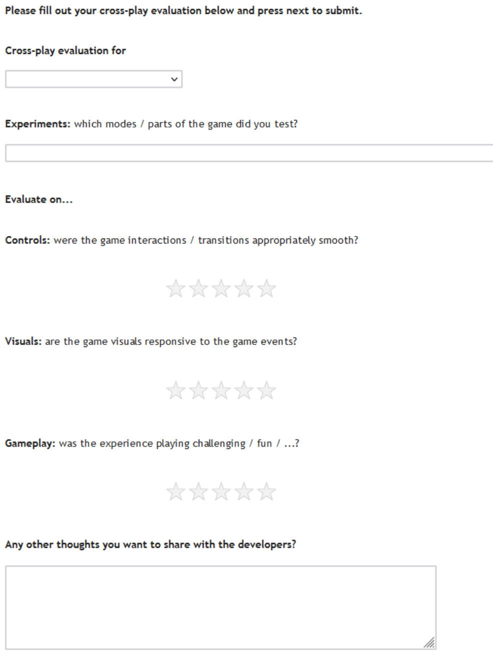
CPSC 427

Cross-Play Evaluation Form

IMPORTANT: Only submit evaluations for other teams games, DO NOT submit evaluations for your teams game. We will consider it cheating if we find anyone submitting evaluations for their own game!

Student name

Student number



Please fill out your cross-play evaluation below and press next to submit.

Cross-play evaluation for

Experiments: which modes / parts of the game did you test?

Evaluate on...

Controls: were the game interactions / transitions appropriately smooth?

☆☆☆☆☆

Visuals: are the game visuals responsive to the game events?

☆☆☆☆☆

Gameplay: was the experience playing challenging / fun / ...?

☆☆☆☆☆

Any other thoughts you want to share with the developers?

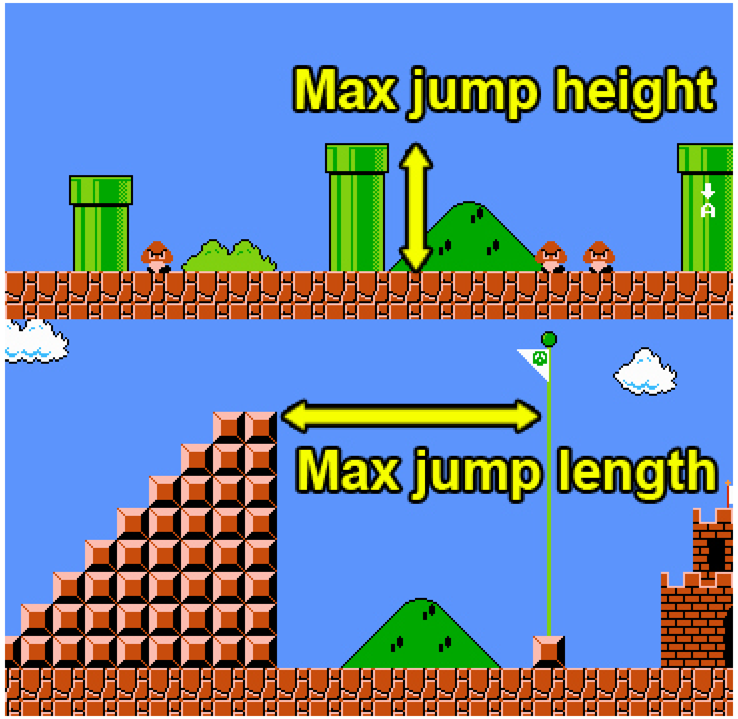
Team Presentations

- *Showcase your game (live or video)*
- *Share one thing that worked well*
- *Share one thing that did not work/took longer*

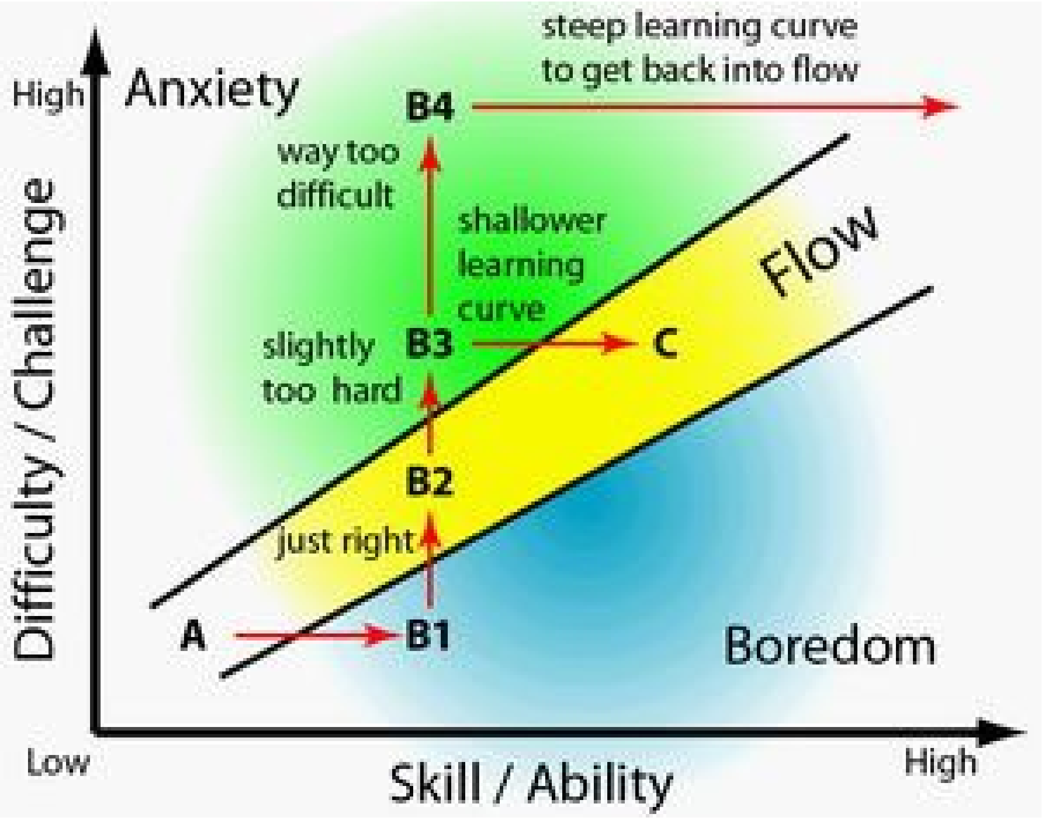
Update slides here:

https://docs.google.com/presentation/d/1vN4NcaJvd015AyxZ0GVFnPeu1EaYMISU0_XAEwsc4ts/edit?usp=sharing

Game Balancing



Fun to play?



https://www.androidauthority.com/level-design-mobile-games-developers-make-games-fun-661877/

Learning goals

- *Know the different aspects of a game that can be balanced.*
- *Connecting game balancing to game theory*
- *Learn about common balancing steps*
- *Practice basic game balancing*

Resources on Balancing

https://gamebalanceconcepts.wordpress.com/2010/07/07/level-1-intro-to-game-balance/

by Ian Schreiber

What does balanced mean?

- *Is chess balanced?*
- *Settlers of Catan?*
- *Is Tetris balanced?*
- *Is your game balanced?*

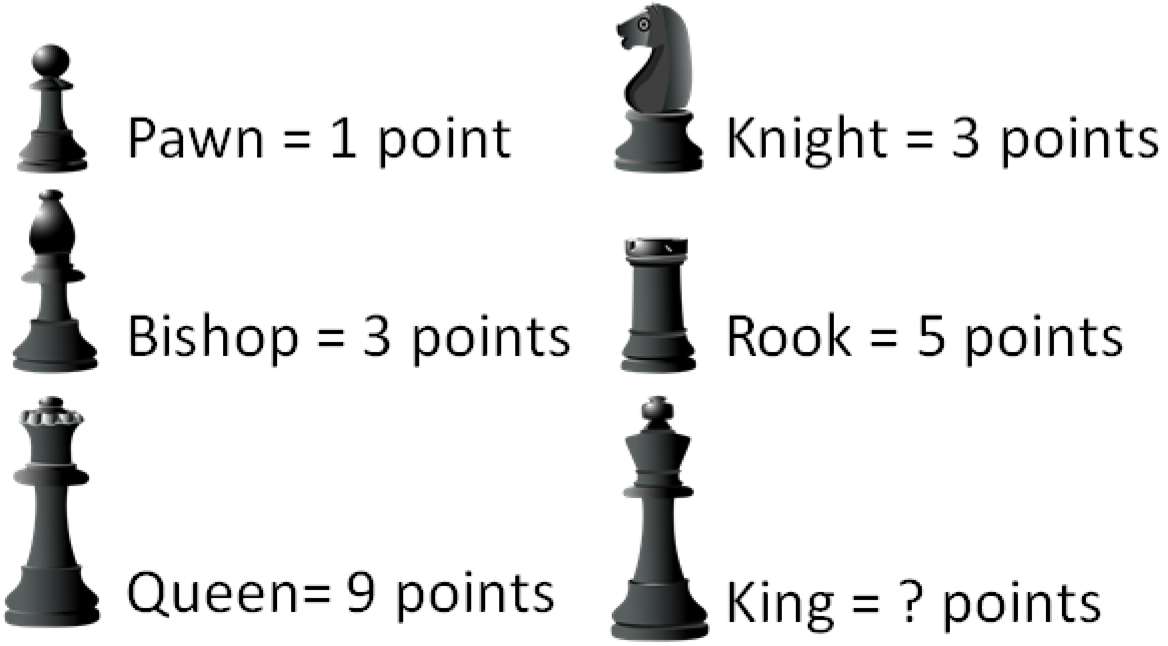
Modeling

Indirect relationships

Value of a piece

- *it is not possible to get a knight for 3 pawns*
- *one can 'trade' pieces*
- *a currency*

How to determine?



Pawn = 1 point

Knight = 3 points

Bishop = 3 points

Rook = 5 points

Queen= 9 points

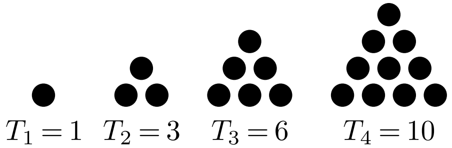
King = ? points

Numeric quantities

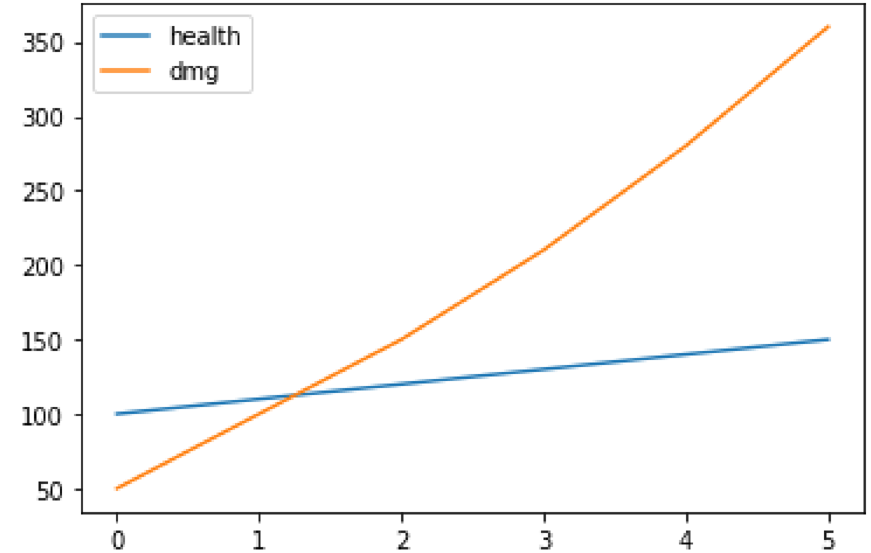
- ***Values in your game?***

Relationships

- ***Linear relations***
- ***Exponential relations***
- ***Triangular relationship***
  - 1, 3, 6, 10, 15, 21, 28, …

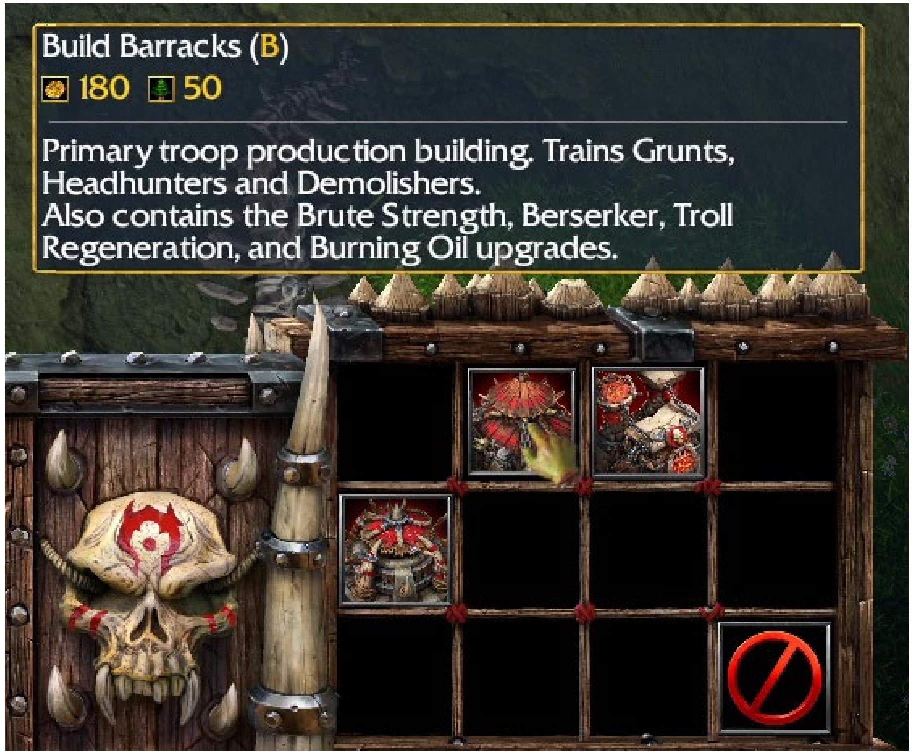


  - The difference increases linearly
  - The function has quadratic complexity
- ***Periodic relations***
  (summer, winter, …)



Model interactions between relationships

- *2x item A + 1x item B = 5x item C*
- *Attack speed * damage = damage / second*
- *Buff: 2x health or +100 health*
  - *what is better?*
- *Progression:*
  - *XP -> level up -> new skill -> ?*



Build Barracks (B)

180 50

Primary troop production building. Trains Grunts, Headhunters and Demolishers.
Also contains the Brute Strength, Berserker, Troll Regeneration, and Burning Oil upgrades.

Game Theory

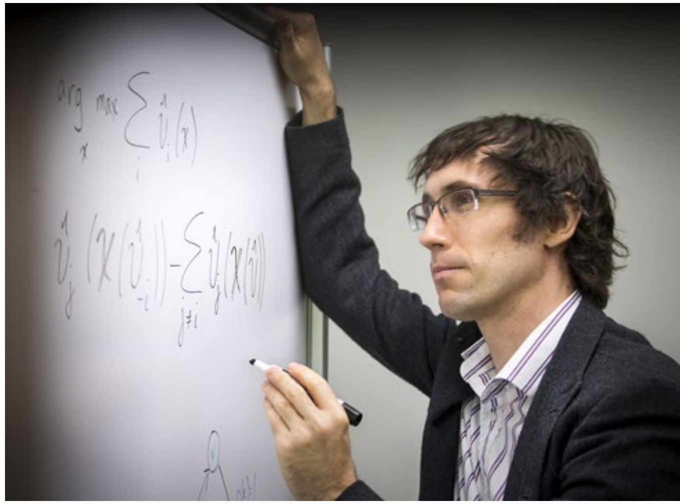
A mathematical concept

- *Used for trading, road design, …*

Terminology

- **Dominant strategy**: one path that is stronger than all others
- **Fairness**: equal chances to win
- **Nash equilibrium:** each player's strategy is optimal when considering the decisions of other players

Interested?



Kevin Leyton-Brown

Important Considerations

- ***Determinism vs. randomness***
- ***Solvability***
  - *Has a best/dominant strategy*
    - Is this desirable?
  - *Can you solve a non-deterministic game?*

- ***Intransitive games***
  - simultaneous choice between opponents, e.g. *Rock-Paper-Scissors*
- ***Symmetric***
  - *same chances*
- ***Game and meta game***

Learning goals

- ***Know the different aspects of a game that can be balanced.***
- ***Connecting game balancing to game theory***
- ***Learn about common balancing steps***
- ***Practice basic game balancing***

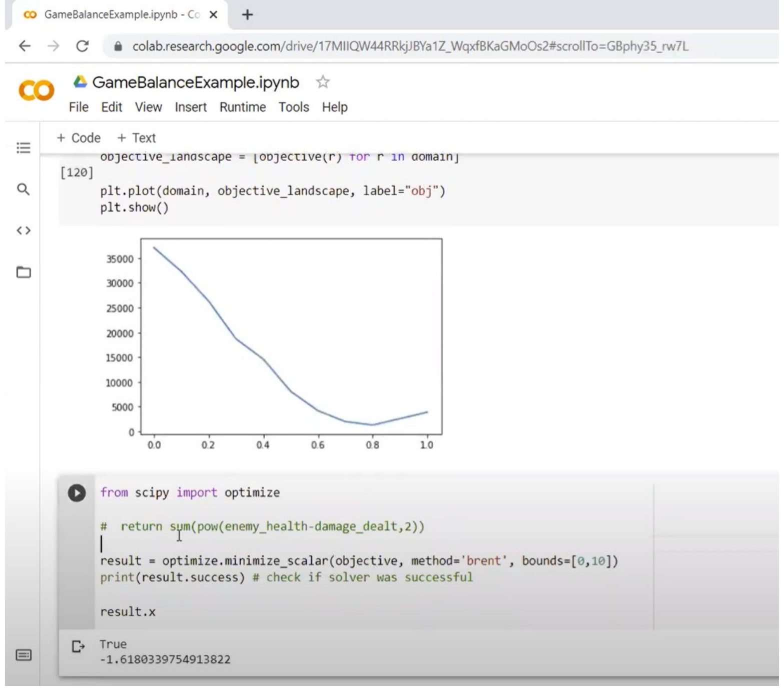
Breakout

- ***List all relevant quantities in your game***
- ***List their relations:***
  - Type: e.g., linear
  - Quantitatively: e.g., +5 gold per round,
    1 gold = 100 silver
- *Investigate interactions between relations*
- ***What is your game's currency?***
  - *Gold, life, …?*

6 min
select representative to report

Self study? Numerical Methods - Optimization

- ***Iterative optimizers***
  - Single variable?
  - Multiple variables?
  - Gradient descent?



```
    objective_landscape = [objective(r) for r in domain]
[120]
    plt.plot(domain, objective_landscape, label="obj")
    plt.show()
```

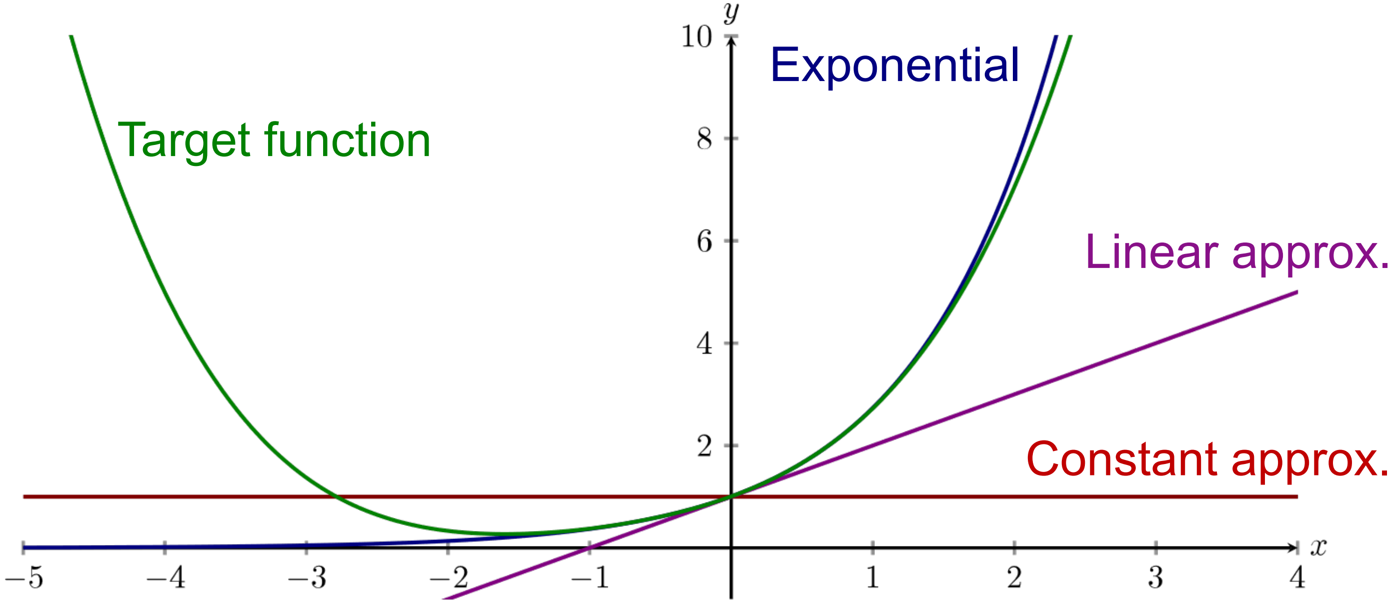
```
from scipy import optimize

#  return sum(pow(enemy_health-damage_dealt,2))

result = optimize.minimize_scalar(objective, method='brent', bounds=[0,10])
print(result.success) # check if solver was successful

result.x
```

```
True
-1.6180339754913822
```



Lecture: https://youtu.be/ZNsNZOnrM50

- **Balancing demo starts at 1h20**
- **Optimizer used at ~ 1h30**

Learning goals

- *Know the different aspects of a game that can be balanced.*
- *Connecting game balancing to game theory*
- *Learn about common balancing steps*
- *Practice basic game balancing*

Balancing example

- *10 enemies per level*
- *One tower does 1 damage / sec*
- *One tower costs 2 gold*
- *It takes enemies 10 seconds to pass*

- ***How much gold should the player start with?***

- ***Enemy health increases:*** *11,12,14,17,21,…*
- ***How much gold should the player get in round 2?***
- ***How much gold should each eliminated enemy give?***

Demo

Difficulties:

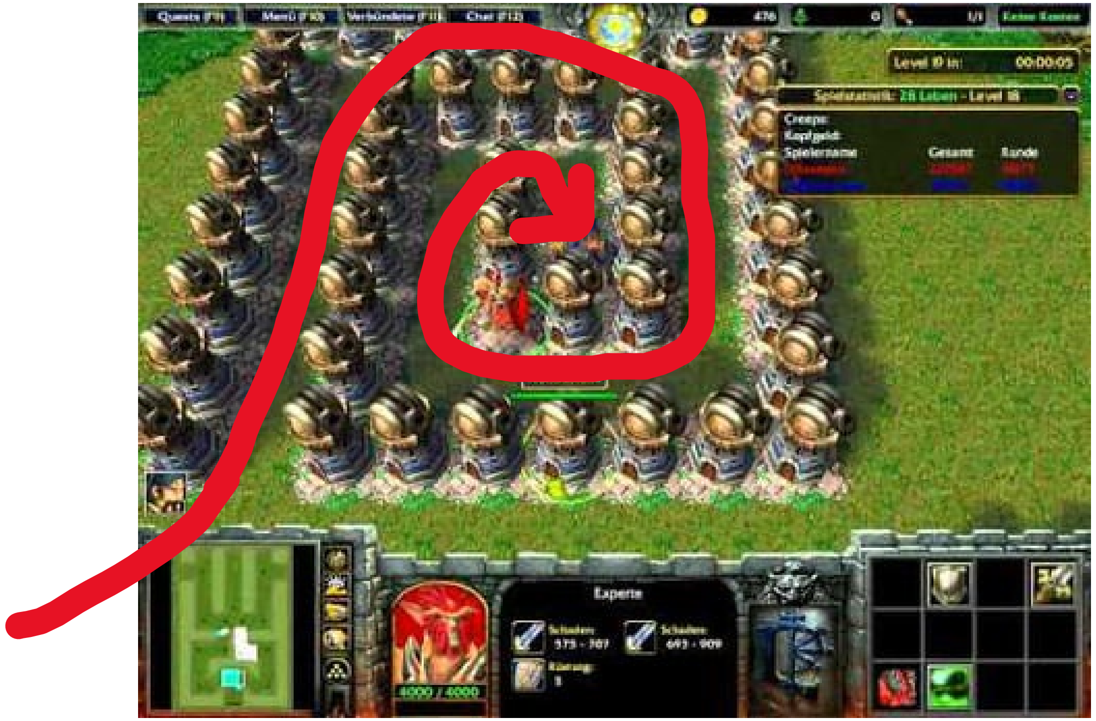
- ***Placement of towers changes the time damage is dealt***

- ***Path of enemies can be hindered to increase time***

➢ ***Measure during playtest***

  ➢ *cross-play*

- ***Some enemies are resistant to fire/magic/…?***
  - *kind of a periodic feature*



Learning goals

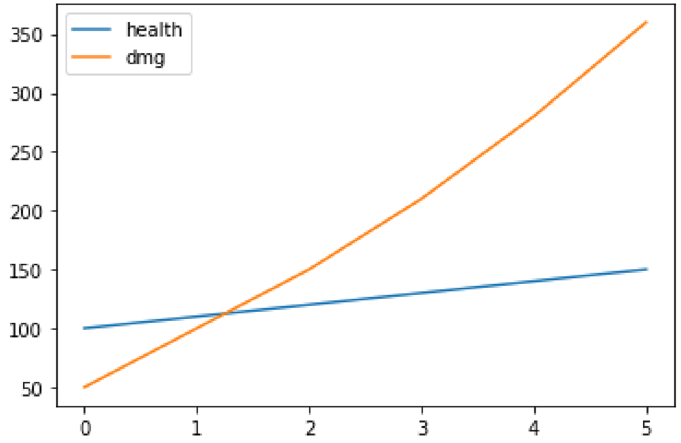
- *Know the different aspects of a game that can be balanced.*
- *Connecting game balancing to game theory*
- *Learn about common balancing steps*
- *Practice basic game balancing*

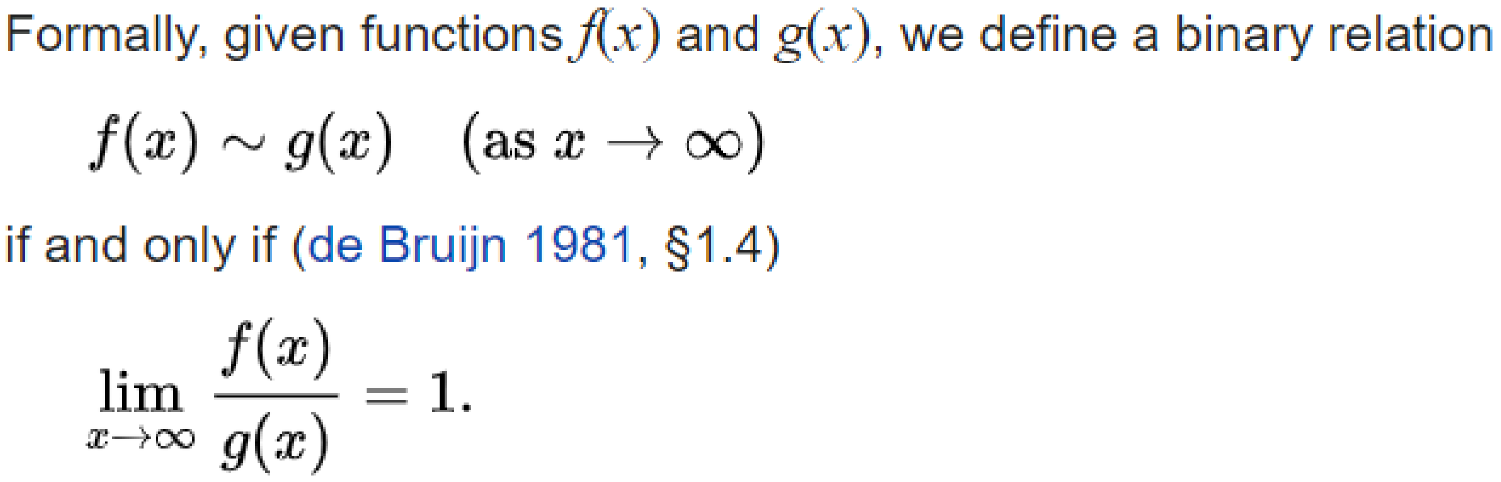
Counter Measures

- **Transitive Mechanics**
  - *Repair costs*
  - *Consumables (food, potions, …)*
  - *Tax*

Asymptotic analysis?

- *Linear * linear?*
- *Linear + linear?*
- *Linear + exponential?*
- *Linear * exponential?*





Formally, given functions $f(x)$ and $g(x)$, we define a binary relation

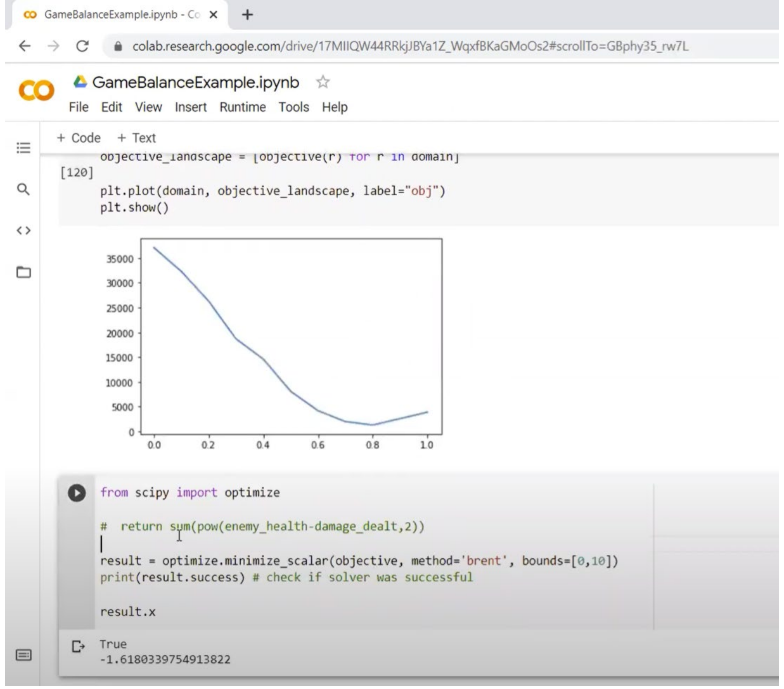
$$f(x) \sim g(x) \quad (\text{as } x \to \infty)$$

if and only if (de Bruijn 1981, §1.4)

$$\lim_{x \to \infty} \frac{f(x)}{g(x)} = 1.$$

Numerical Methods - Optimization

- ***Iterative optimizers***
  - Single variable?
  - Multiple variables?
  - Gradient descent?



```
    objective_landscape = [objective(r) for r in domain]
[120]
    plt.plot(domain, objective_landscape, label="obj")
    plt.show()
```

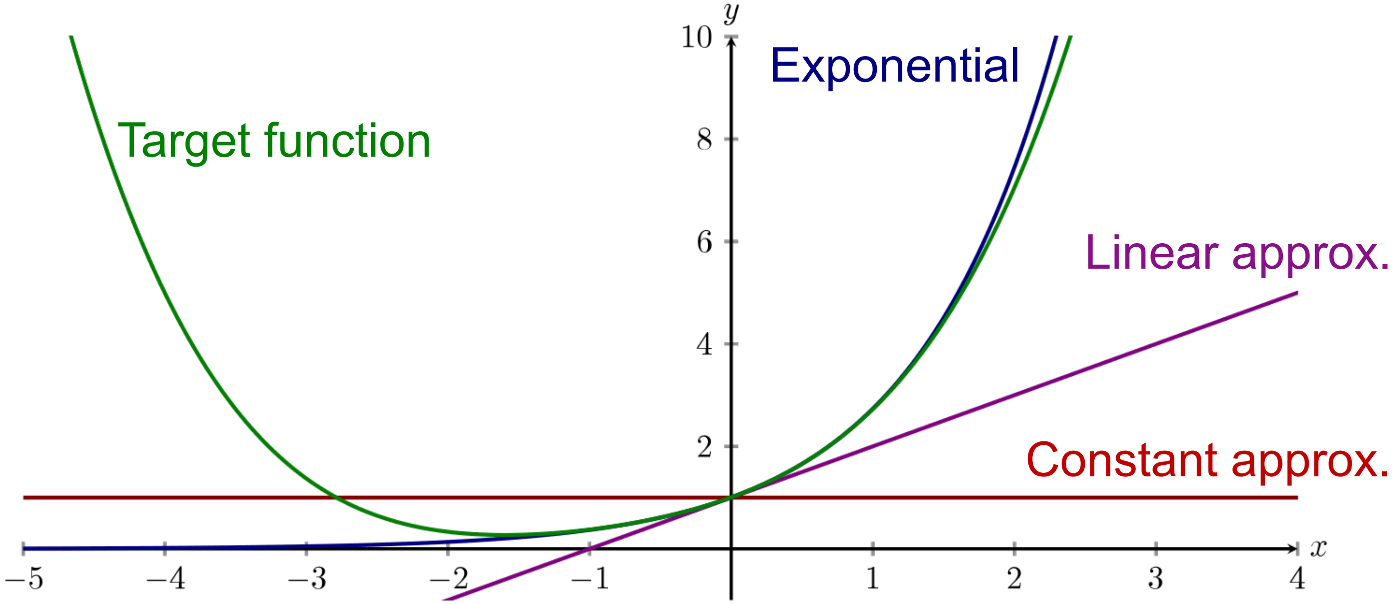
```
from scipy import optimize

#  return sum(pow(enemy_health-damage_dealt,2))

result = optimize.minimize_scalar(objective, method='brent', bounds=[0,10])
print(result.success) # check if solver was successful

result.x
```

```
True
-1.6180339754913822
```



Lecture: https://youtu.be/ZNsNZOnrM50

- **Balancing demo starts at 1h20**
- **Optimizer used at ~ 1h30**

How to quantify difficulty?

- ***Player vs. enemy strength***
- ***Likelihood of winning***
- ***Required skill***
  - *Knowledge of the game*
  - *Reaction*
  - *Precision*
  - *Tactics*

-> Estimate player strength in relation skill level: beginner, intermediate, pro?
-> requires user studies!

© Alla Sheffer, Helge Rhodin

M4 balancing requirement

Carefully balance one aspect of your game (e.g., movement-speed, health points, strength, bonus,…).

- ***Report on the theoretical analysis***
- ***Change log with testing results (before/after balancing)***

Breakout II

- ***Sketch progression***
  - *Quantities over time*
  - *Interactions between quantities*

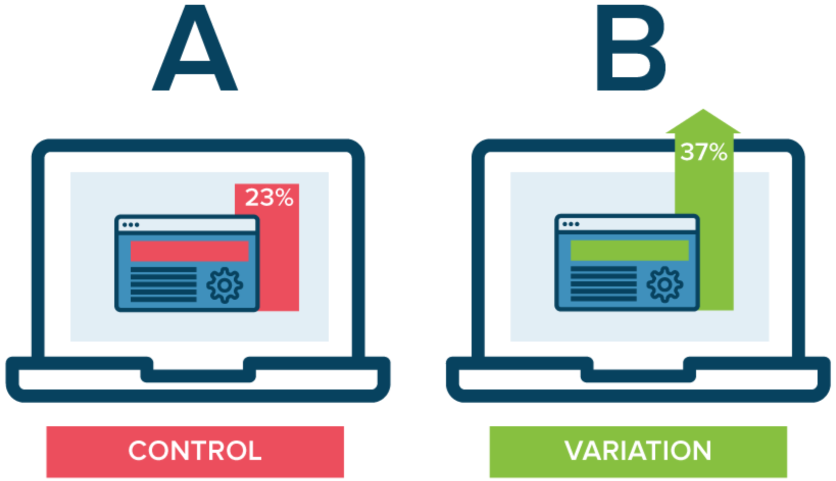
- ***Use pen & paper, plotting tool, or python***

- ***Start balancing your game***

A/B Testing

Testing two variants of your game (with and w/o a feature)

- *randomized participants (same pool)*
- *with respect to a measurable objective (e.g., clicks on website)*



Related to

- two-sample hypothesis testing
- Clinical tests, e.g., testing of a COVID-19 vaccine
- Placebo effect

https://www.optimizely.com/optimization-glossary/ab-testing/