Distinguished Lecture

CS @ Iowa State University

November 11, 2019

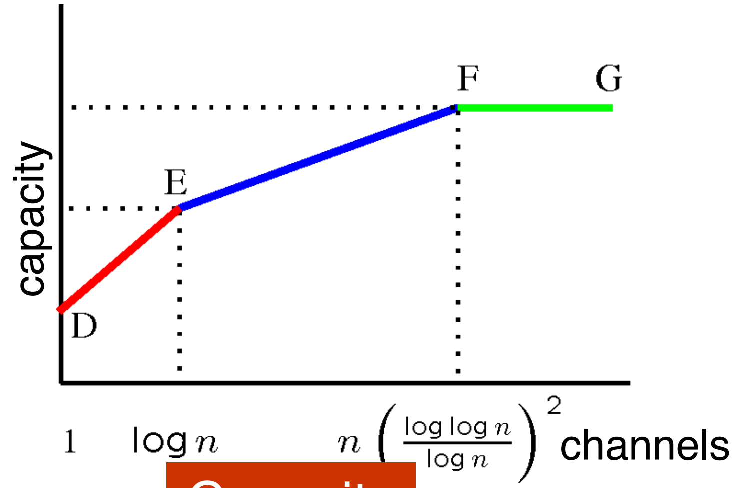
Security and Privacy for Distributed Optimization and Learning

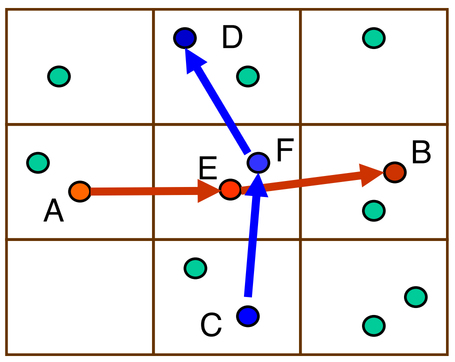
Nitin Vaidya

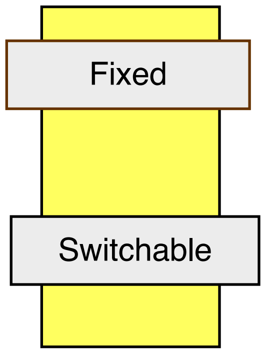
Georgetown University

Net-X: Multi-Channel Mesh

Theory to Practice







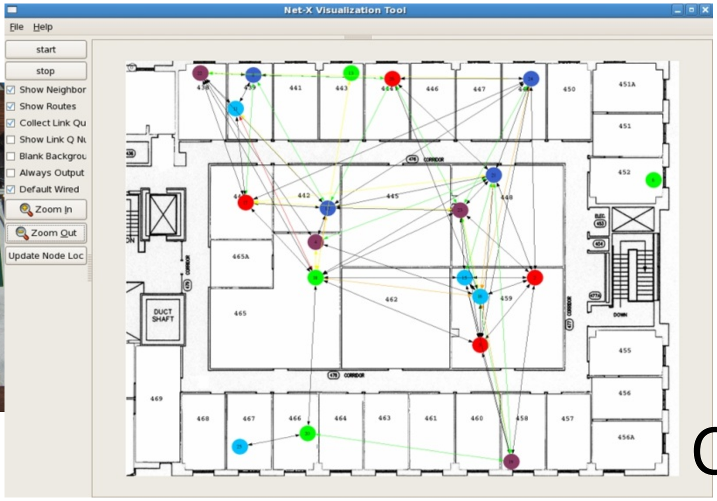
Capacity bounds

Insights on protocol design

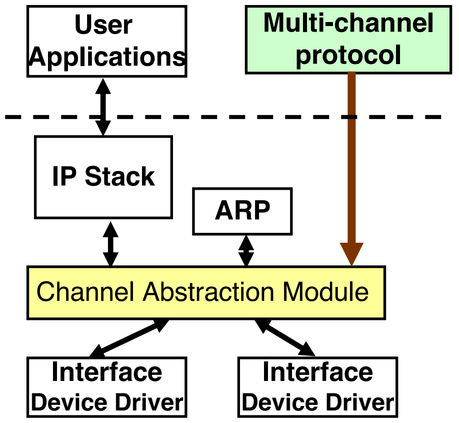
OS improvements Software architecture

Net-X testbed

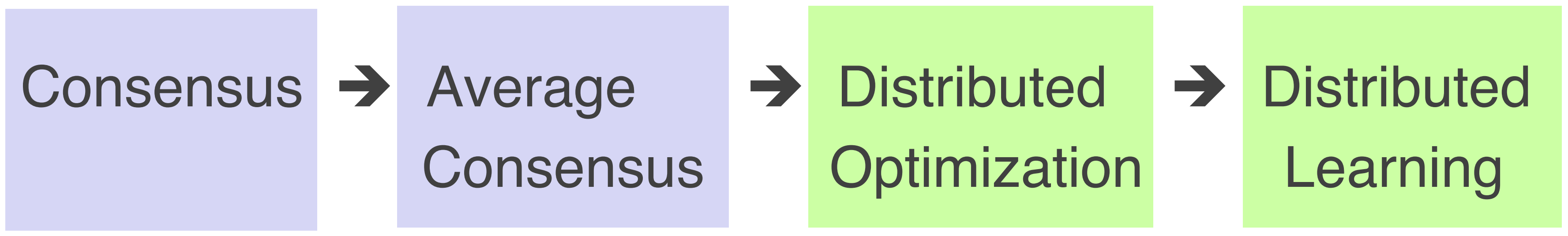
Linux box



CSL



Brief History



Decentralized
Control/Optimization

Distributed
Computing

Security and Privacy for Distributed Optimization and Learning

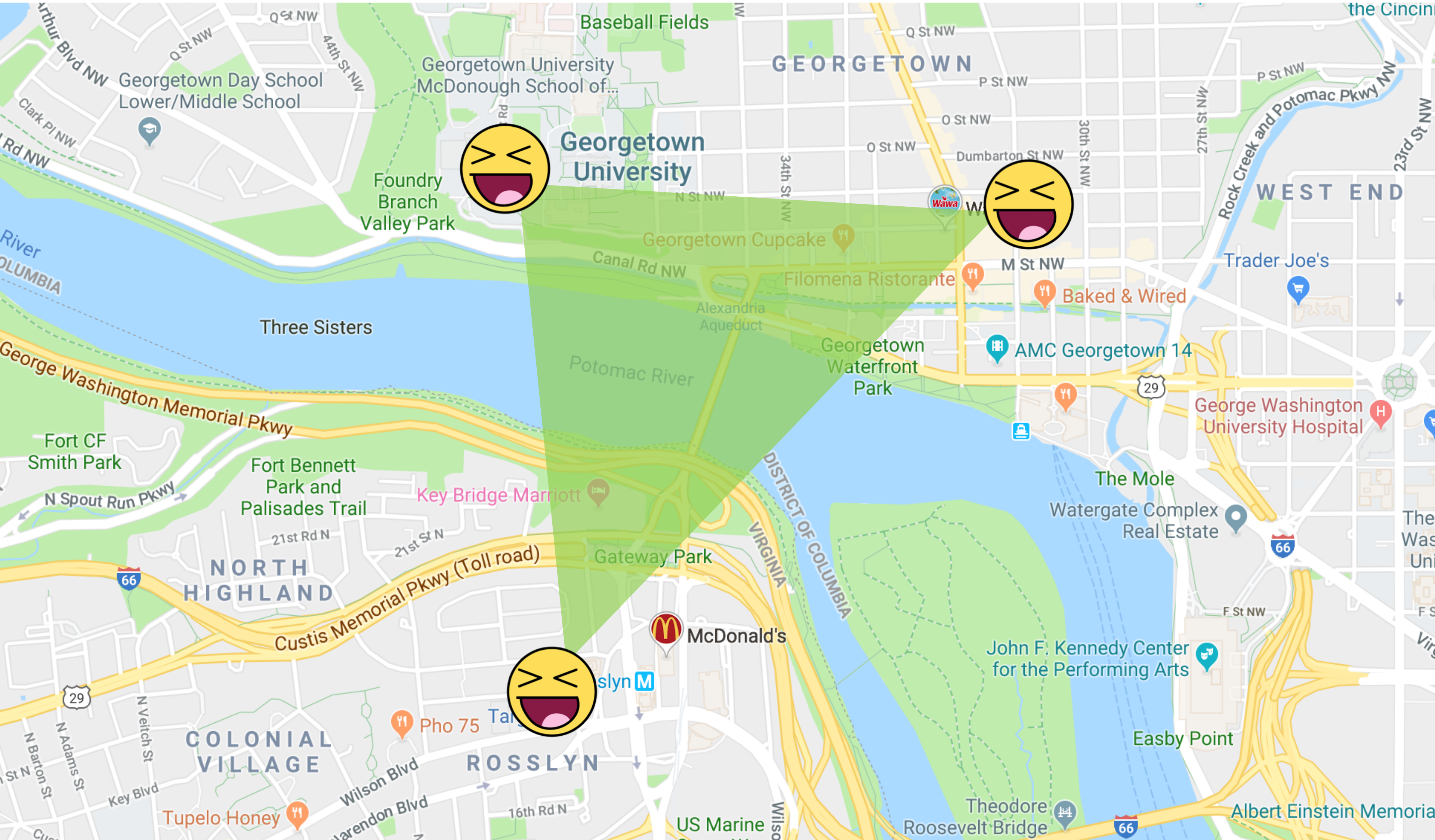
Secret to happiness is to
lower your expectations to the point
where they're already met

- Hobbes (paraphrased)

Goals

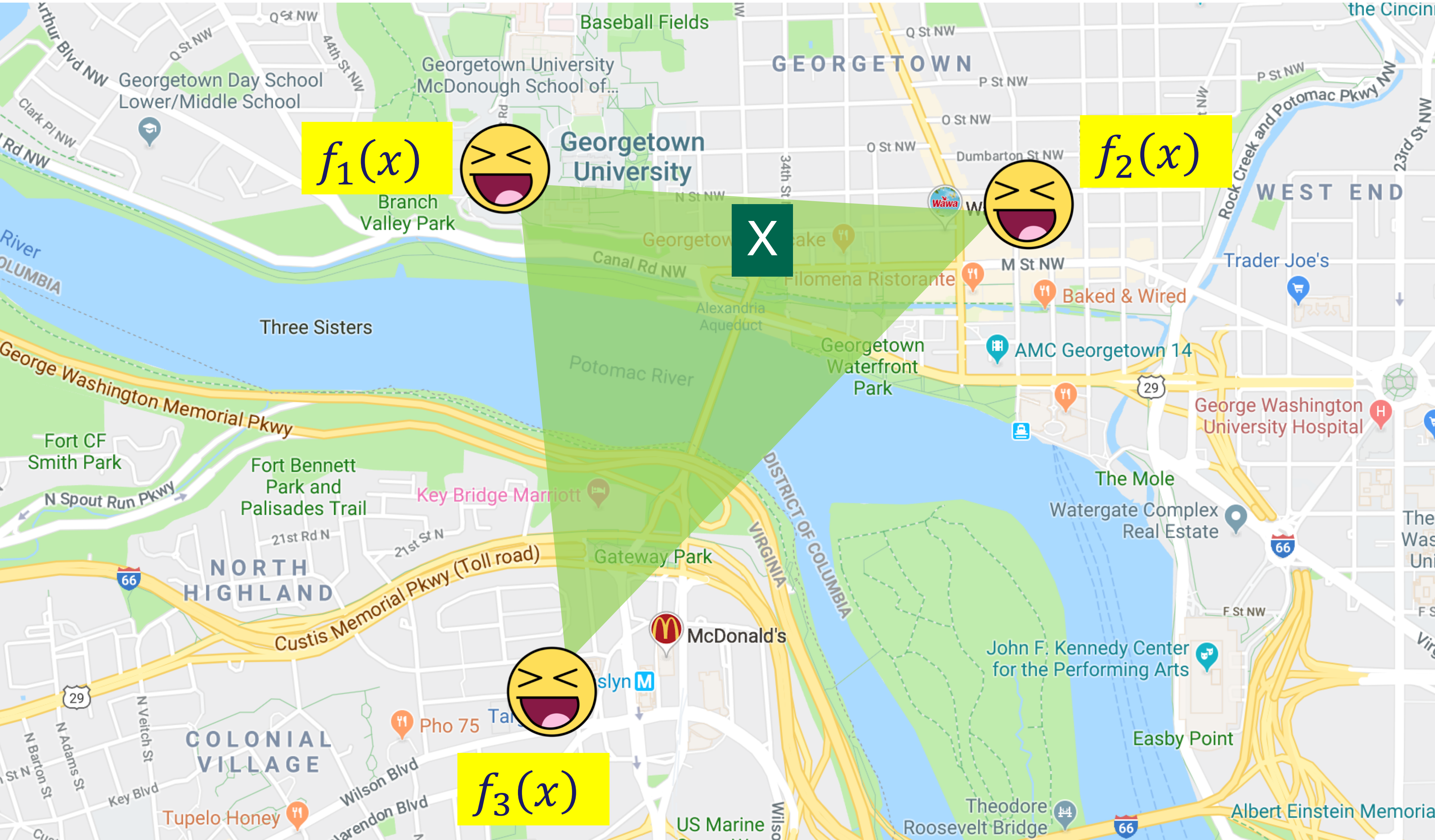
- Background
- Problem formulation
- Intuition
- No theorems / proofs

Rendezvous



GEORGETOWN
Georgetown University
Georgetown University McDonough School of...
Georgetown Day School Lower/Middle School
Foundry Branch Valley Park
Baseball Fields
Georgetown Cupcake
Filomena Ristorante
Baked & Wired
Trader Joe's
WEST END
M St NW
Canal Rd NW
Alexandria Aqueduct
Georgetown Waterfront Park
AMC Georgetown 14
Three Sisters
Potomac River
George Washington Memorial Pkwy
Fort CF Smith Park
Fort Bennett Park and Palisades Trail
Key Bridge Marriott
Gateway Park
NORTH HIGHLAND
Custis Memorial Pkwy (Toll road)
McDonald's
DISTRICT OF COLUMBIA
VIRGINIA
George Washington University Hospital
The Mole
Watergate Complex Real Estate
John F. Kennedy Center for the Performing Arts
Easby Point
Theodore Roosevelt Bridge
Albert Einstein Memoria
COLONIAL VILLAGE
ROSSLYN
Pho 75
Tupelo Honey
Wilson Blvd
Clarendon Blvd
US Marine
Rock Creek and Potomac Pkwy NW

Rendezvous

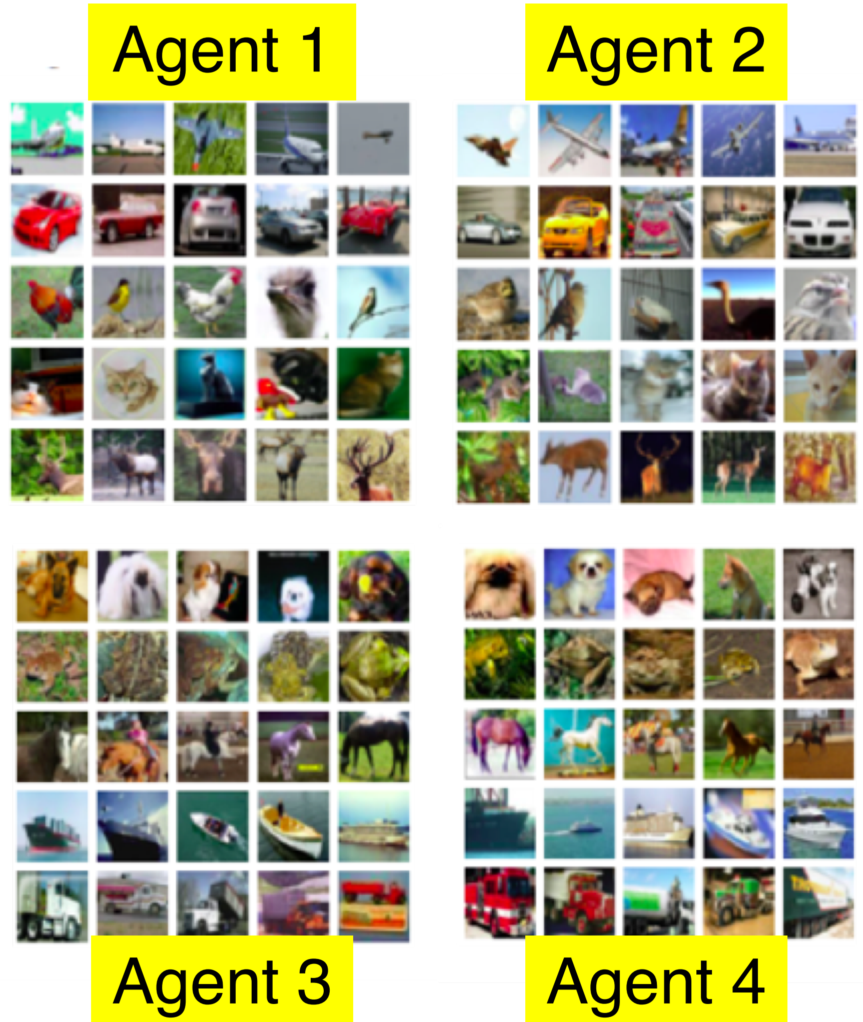


$f_1(x)$
$f_2(x)$
$f_3(x)$
X
Georgetown University
GEORGETOWN
WEST END
Potomac River

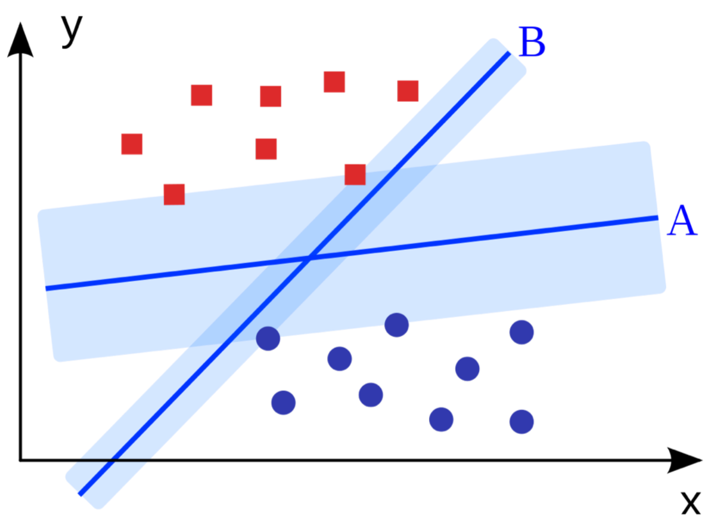
$$\operatorname{argmin} \sum f_i(x)$$

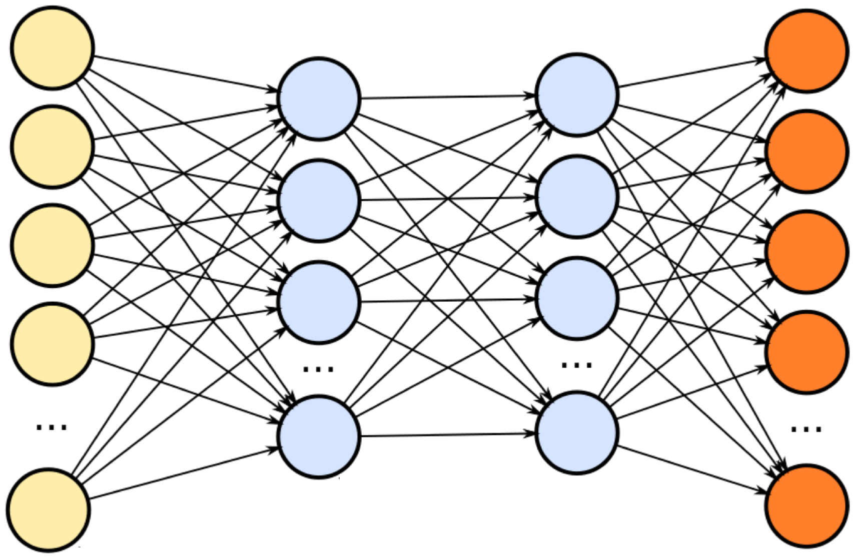
Machine Learning

- Data is distributed across different agents



- Data is distributed across different agents ➔ Collaborate to learn



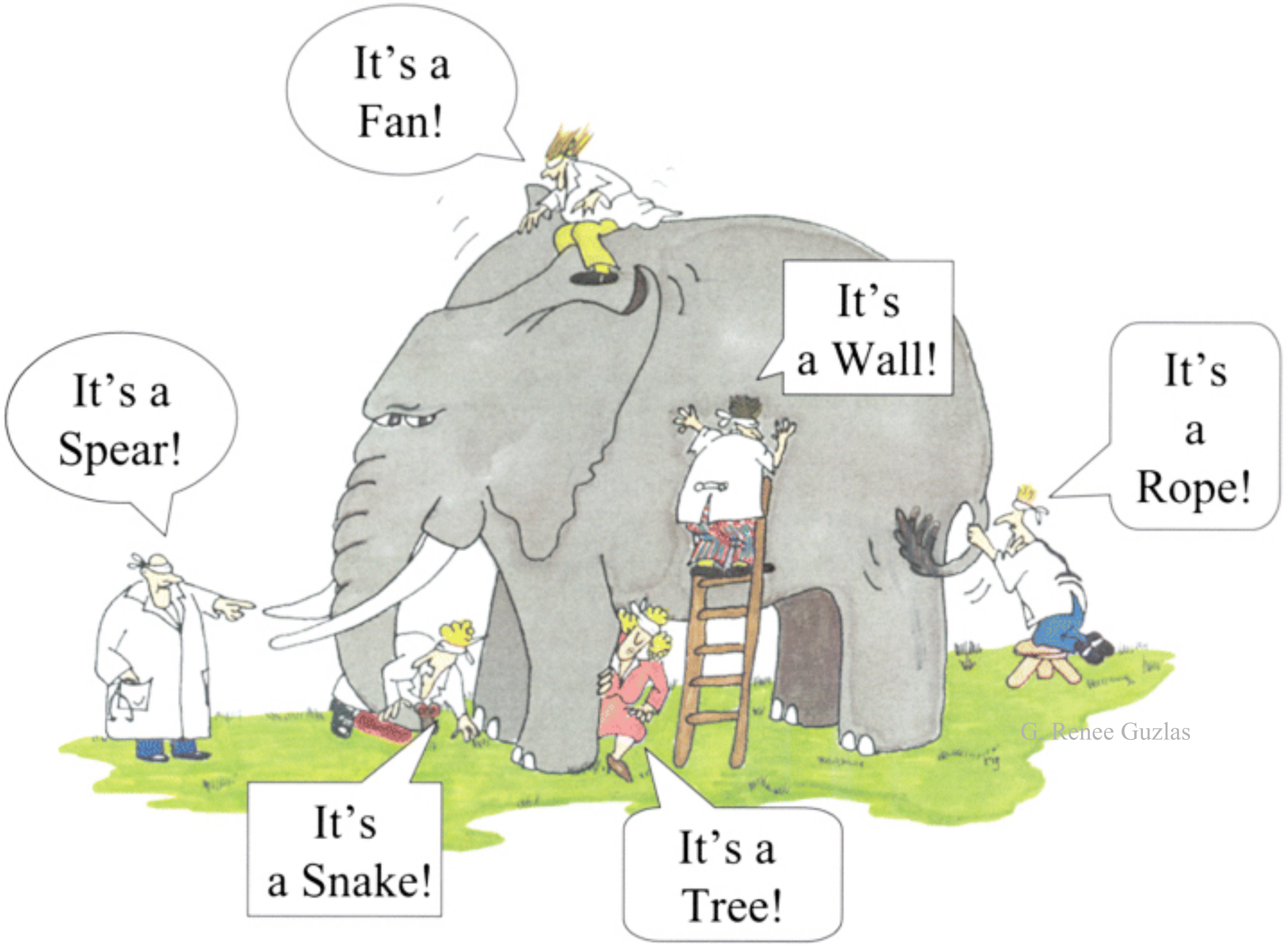


Machine Learning

Minimize global loss

$$\sum f_i(x)$$

Classification

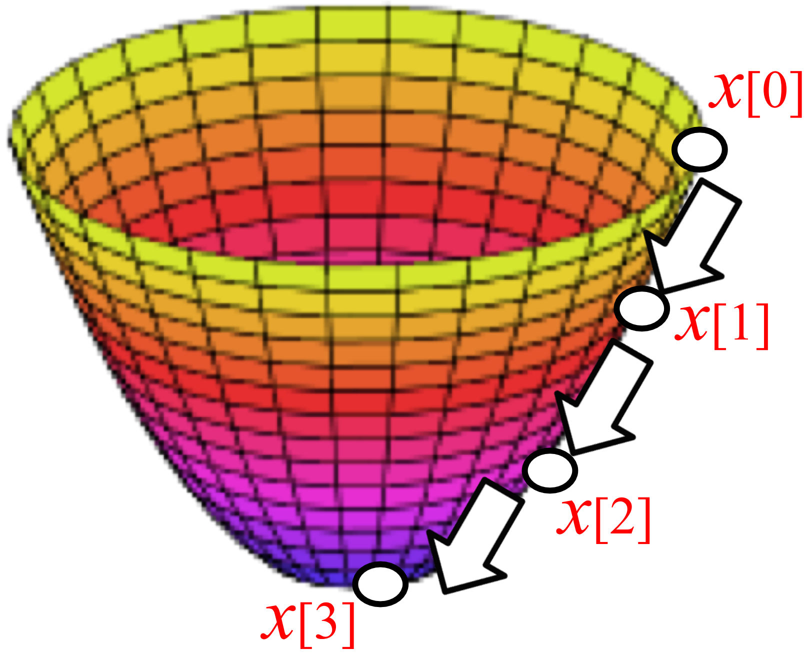


Many Applications

$$\operatorname{argmin} \sum f_i(x)$$

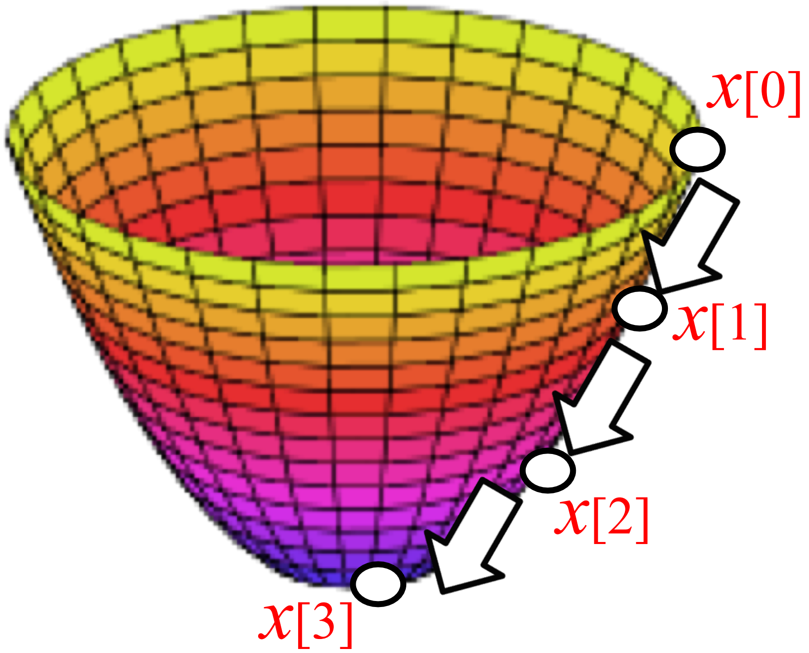
Gradient Method

$$f(x) = \sum f_i(x)$$



Gradient Method

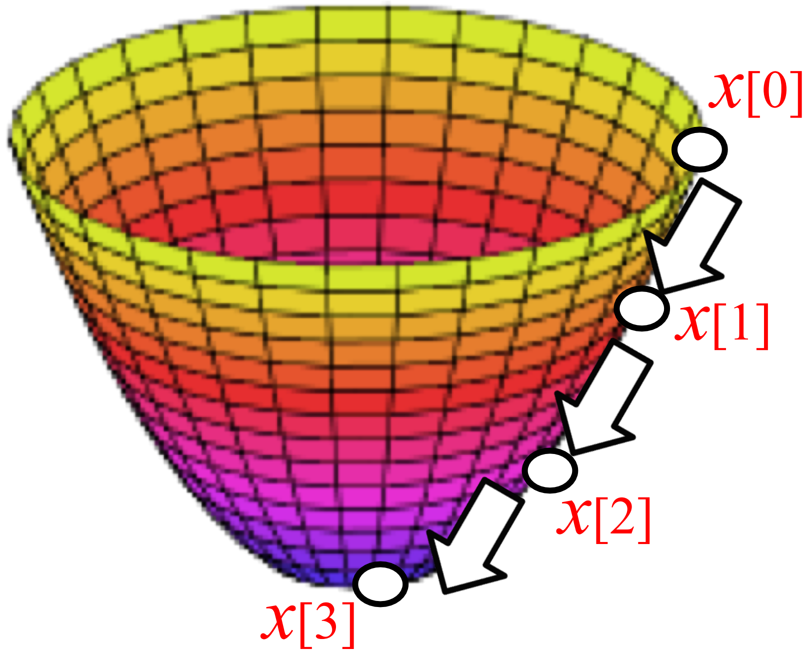
$$f(x) = \sum f_i(x)$$



$$x[k+1] \leftarrow x[k] - \lambda \sum_i \nabla f_i(x[k])$$

Gradient Method

$$f(x) = \sum f_i(x)$$



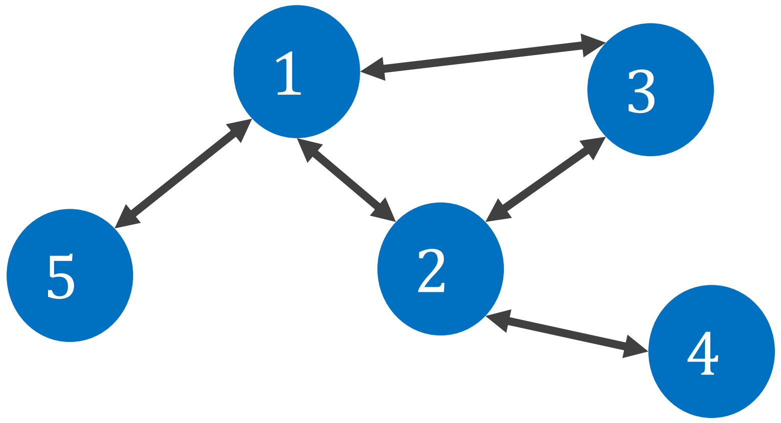
$$x[k+1] \leftarrow x[k] - \lambda \sum_i \nabla f_i(x[k])$$

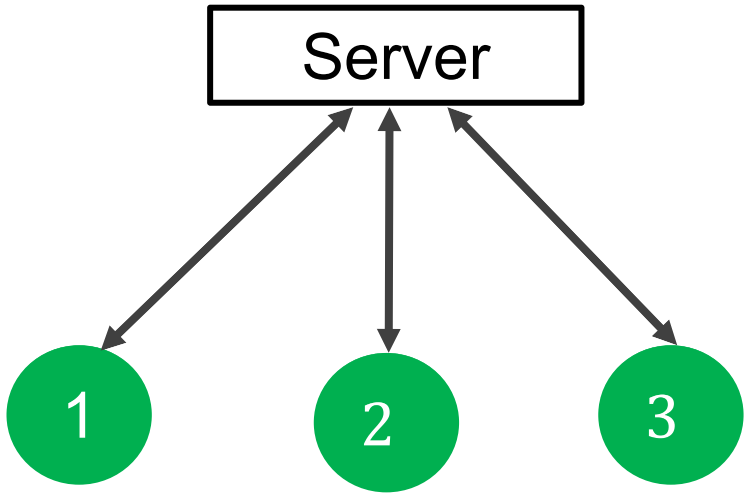
Distributed Optimization

- Each agent i knows own cost function $f_i(x)$
- Need to cooperate to minimize $\sum f_i(x)$

➔ Distributed algorithms

Architectures

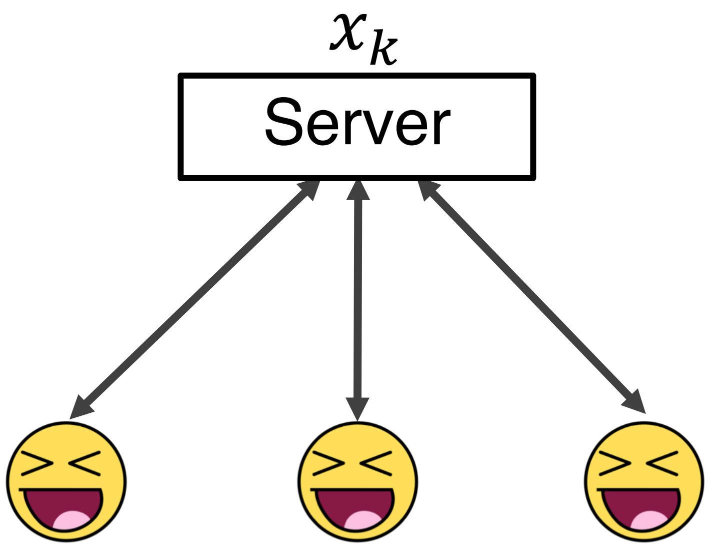




Parameter Servers

Parameter Server

- Server maintains estimate x_k

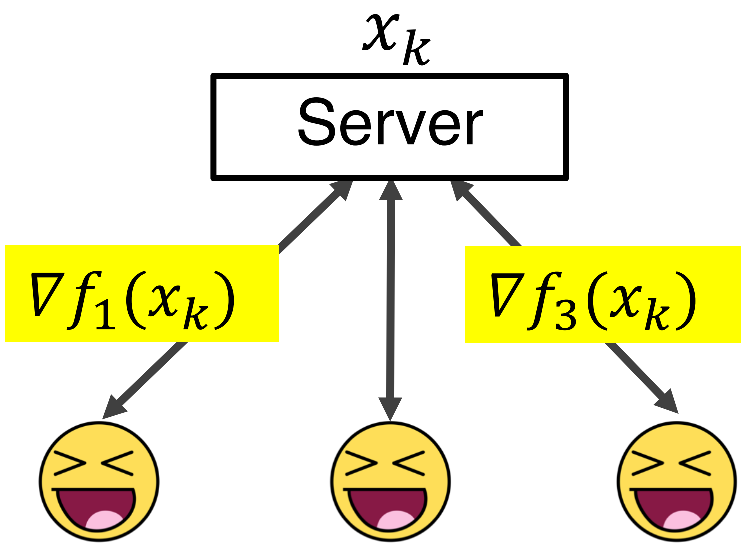


Parameter Server

- Server maintains estimate x_k

<u>In each iteration</u>

- Agent i
    - Receives x_k from server

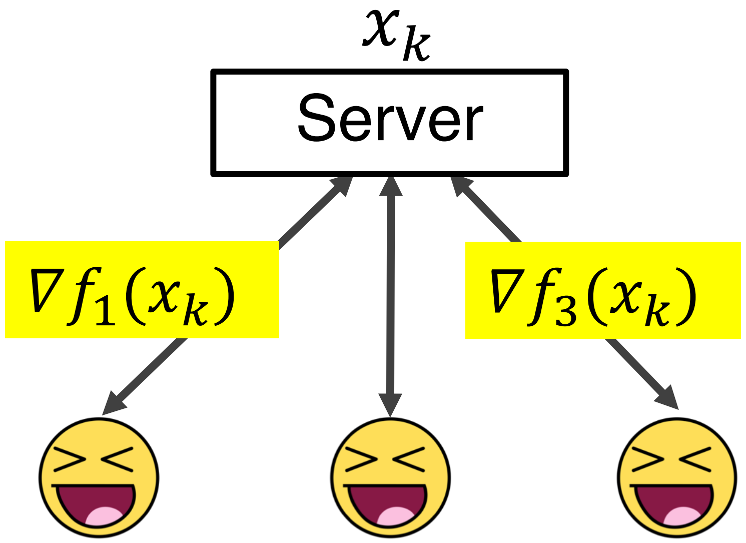


Parameter Server

- Server maintains estimate x_k

<u>In each iteration</u>

- Agent i
  - Receives x_k from server
  - Uploads gradient $\nabla f_i(x_k)$



Parameter Server

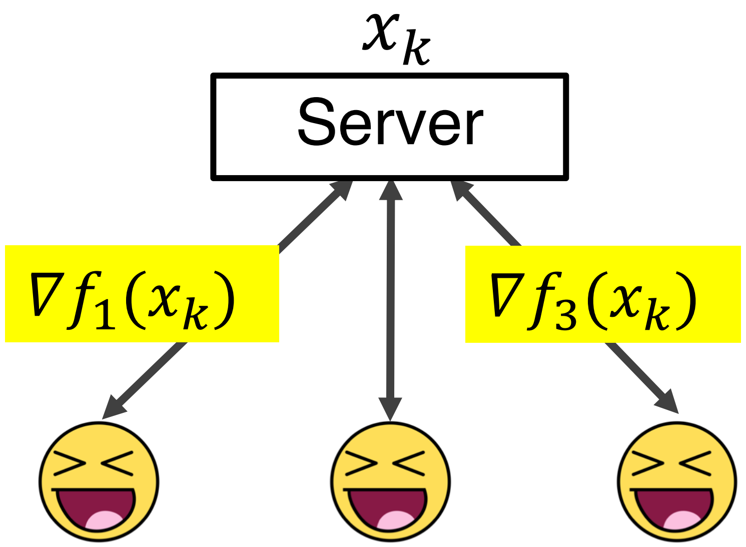
- Server maintains estimate x_k

<ins>In each iteration</ins>

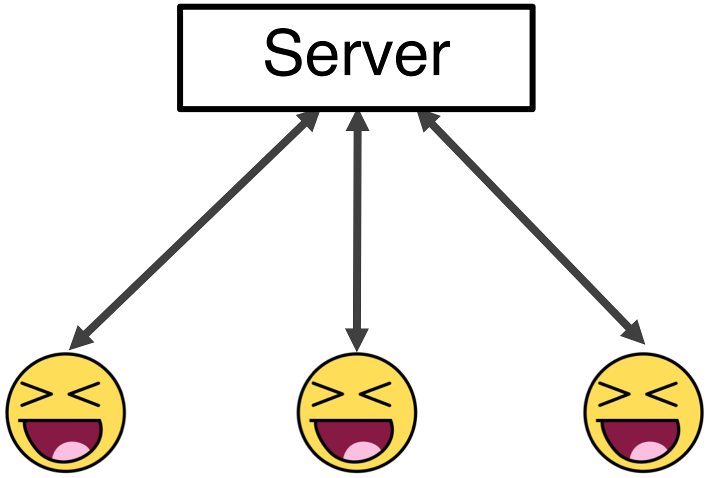
- Agent i
  - Receives x_k from server
  - Uploads gradient $\nabla f_i(x_k)$

- Server updates estimate

$$x_{k+1} \longleftarrow x_k - \alpha \sum \nabla f_i(x_k)$$



Many Variations



... stochastic optimization

... asynchronous

... gradient compression

... acceleration

... shared memory

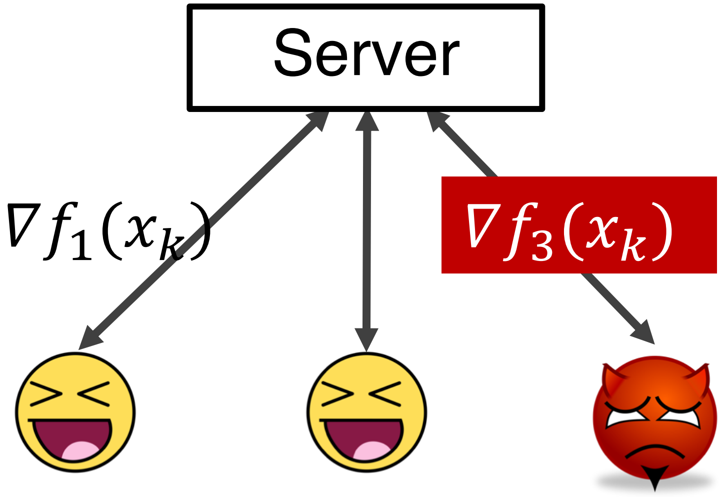
Challenges

Challenges

- Fault-tolerant distributed optimization

$$f_1(x) + f_2(x) + f_3(x)$$

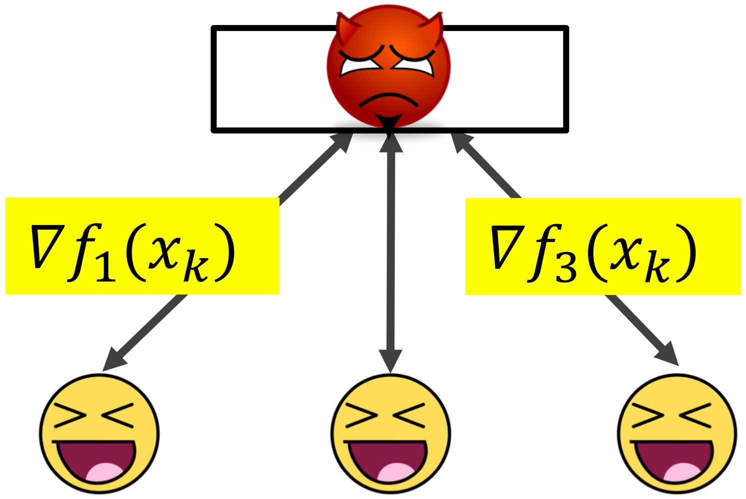
How to optimize if agents inject bogus information?



Challenges

- Privacy-preserving distributed optimization

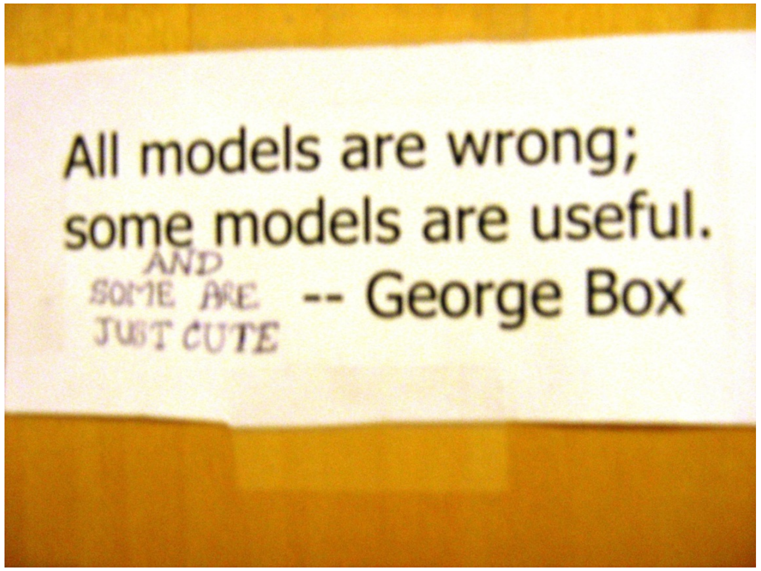
How to collaborate without revealing own cost function?



Fault-Tolerant Optimization

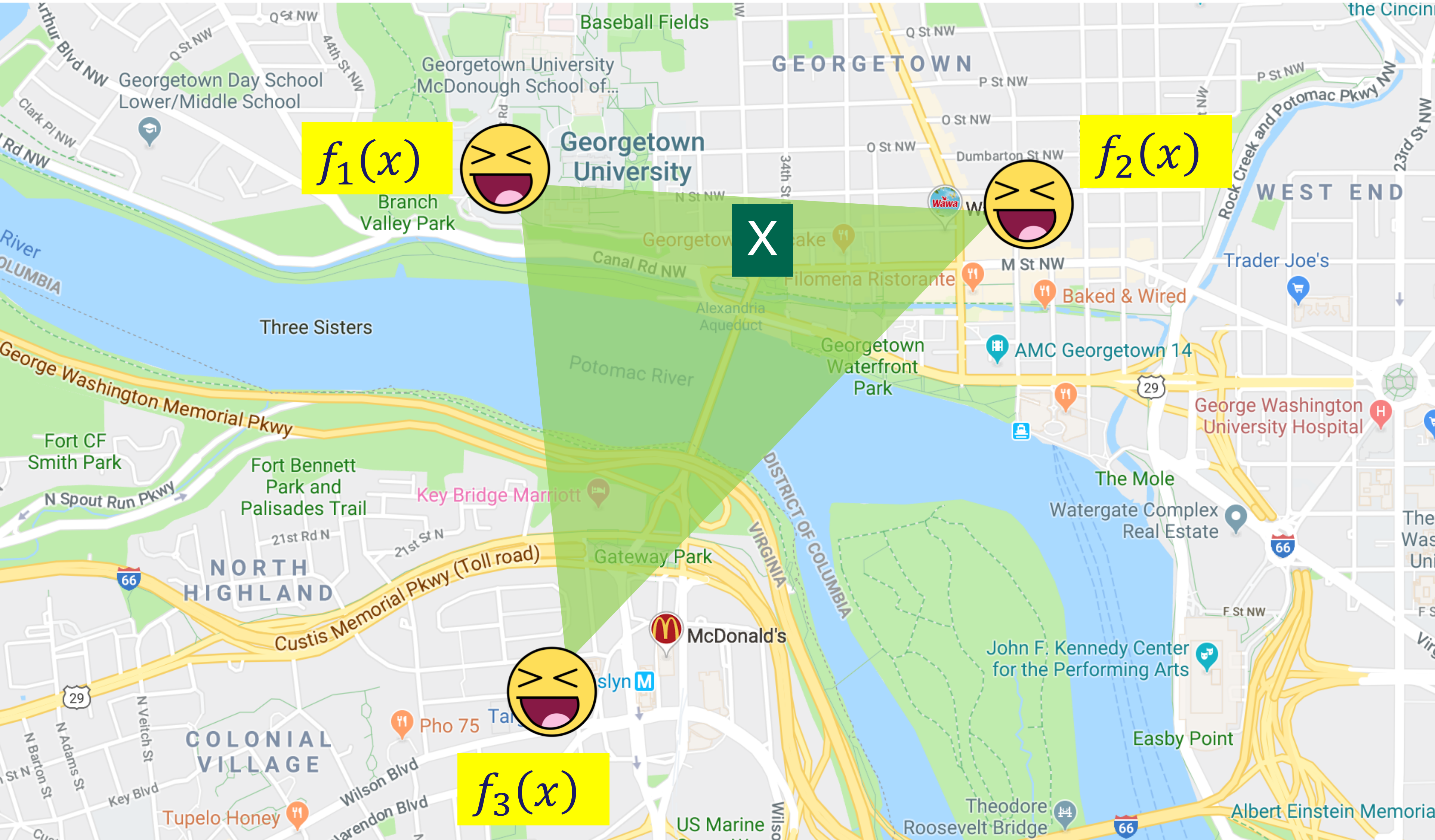
Byzantine Fault Model

- No constraint
  on misbehavior of *faulty* agents



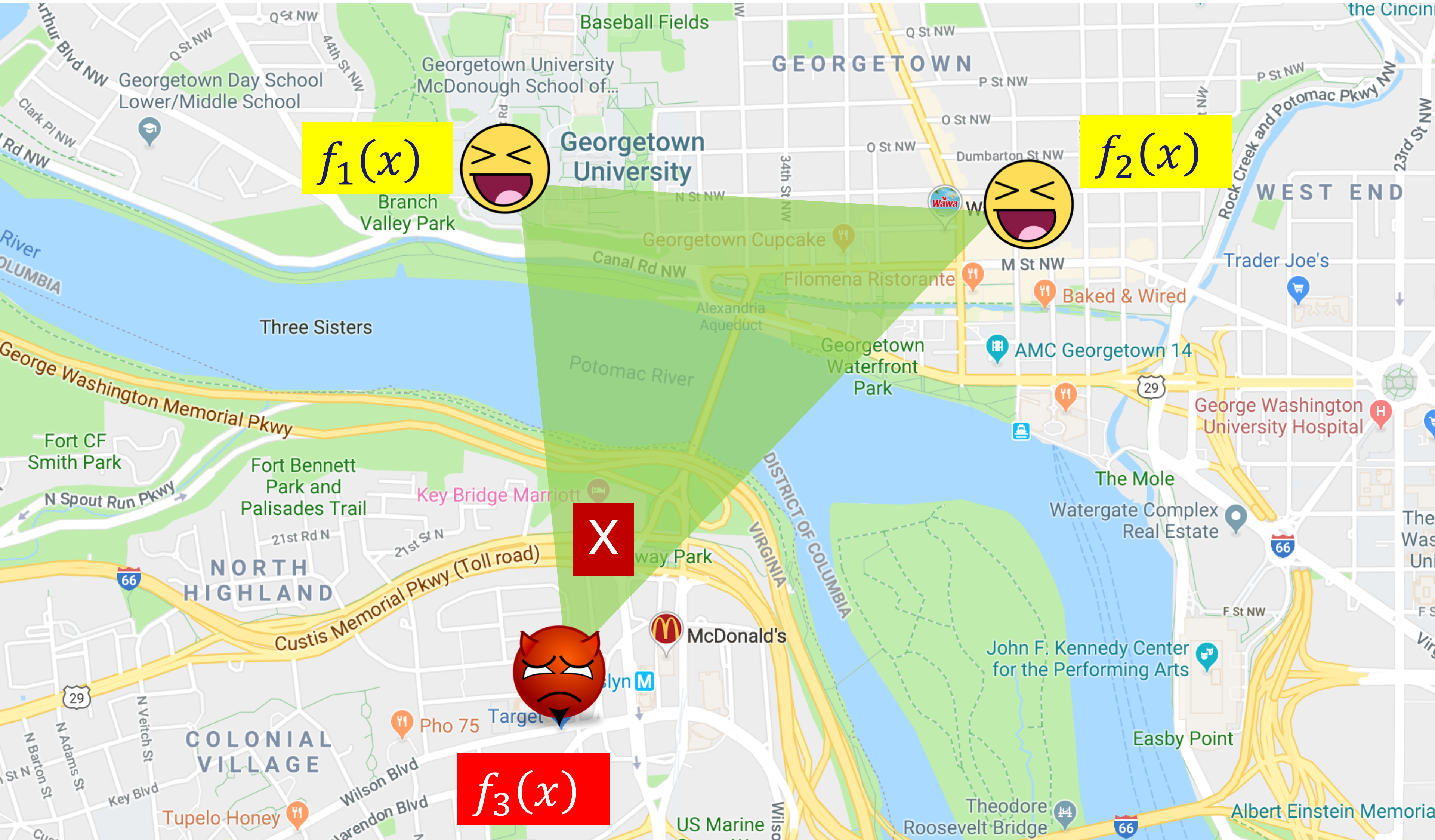
All models are wrong;
some models are useful.
AND SOME ARE JUST CUTE -- George Box

Rendezvous



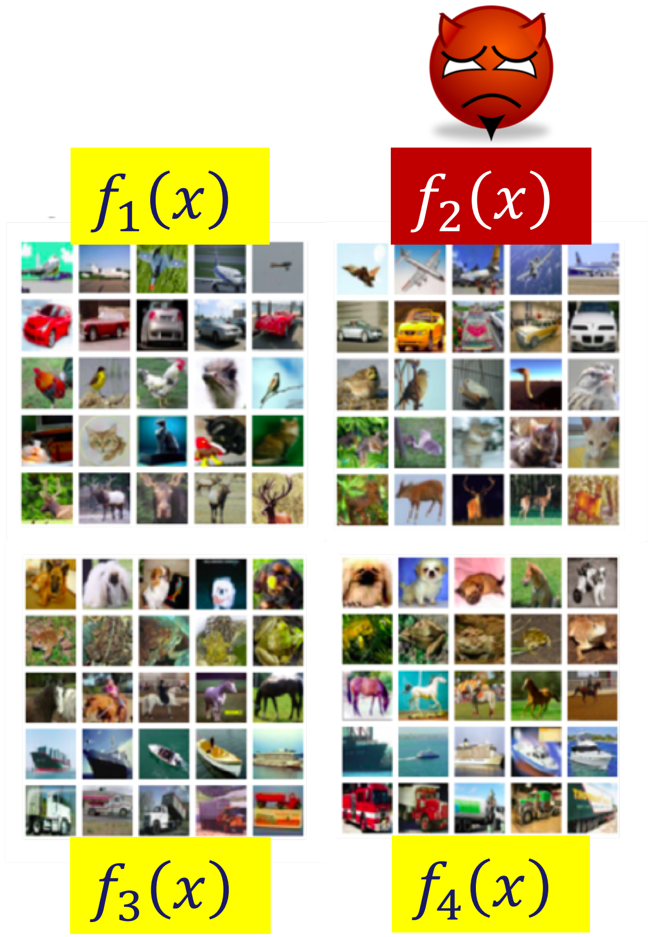
$f_1(x)$
$f_2(x)$
$f_3(x)$
X
Georgetown University
GEORGETOWN
WEST END
Potomac River
NORTH HIGHLAND
COLONIAL VILLAGE

Rendezvous



$f_1(x)$
$f_2(x)$
$f_3(x)$
X
Georgetown University
GEORGETOWN
WEST END
Potomac River
Three Sisters
George Washington Memorial Pkwy
Custis Memorial Pkwy (Toll road)
NORTH HIGHLAND
COLONIAL VILLAGE
Georgetown Waterfront Park
John F. Kennedy Center for the Performing Arts
Theodore Roosevelt Bridge

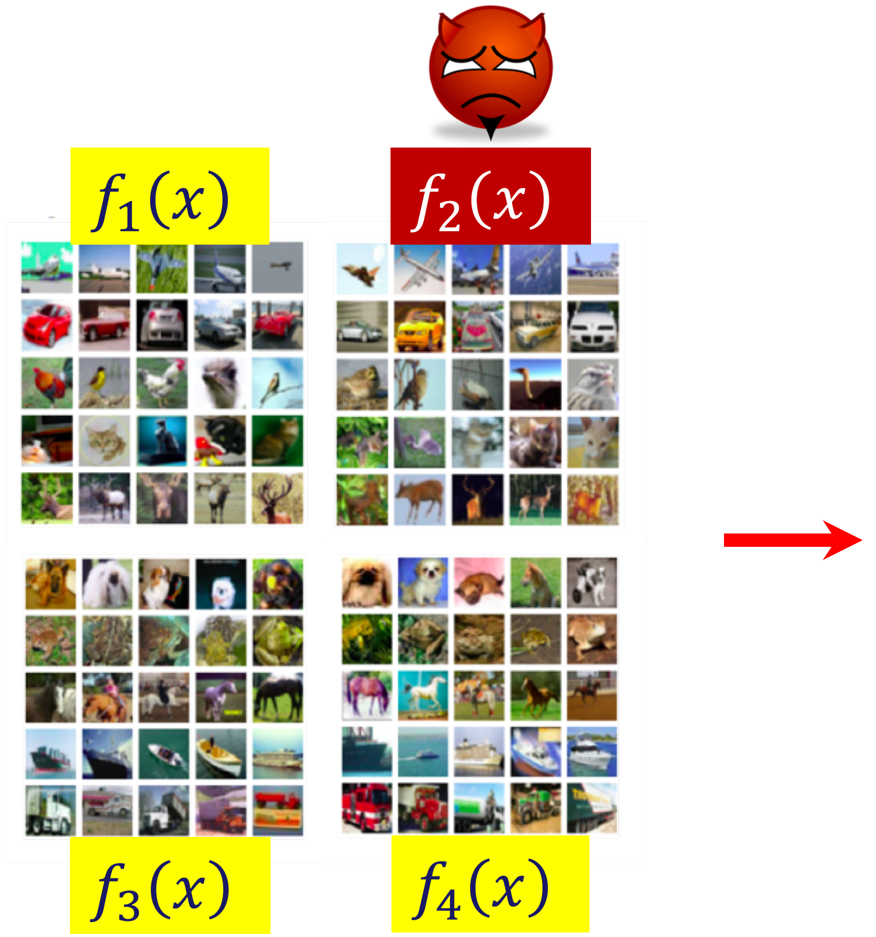
Machine Learning



Minimize global loss

$$\sum f_i(x)$$

- Faulty agent may provide incorrect gradients
- Adversely affect model parameters



Minimize global loss

$$\sum f_i(x)$$

Fault-Tolerance

- What should be the objective of fault-tolerant optimization?

Fault-Tolerance

- What should be the objective of fault-tolerant optimization?

- Optimize over only non-faulty agents … set N

Fault-Tolerance

- What should be the objective of fault-tolerant optimization?

- Optimize over only non-faulty agents … set *N*

$$\operatorname{argmin} \sum_{i \in N} f_i(x)$$

$$\operatorname{argmin} \sum_{i \in N} f_i(x)$$

Is this achievable?

$$\operatorname{argmin} \sum_{i \in N} f_i(x)$$

Is this achievable?

It Depends

$$\operatorname{argmin} \sum_{i \in N} f_i(x)$$

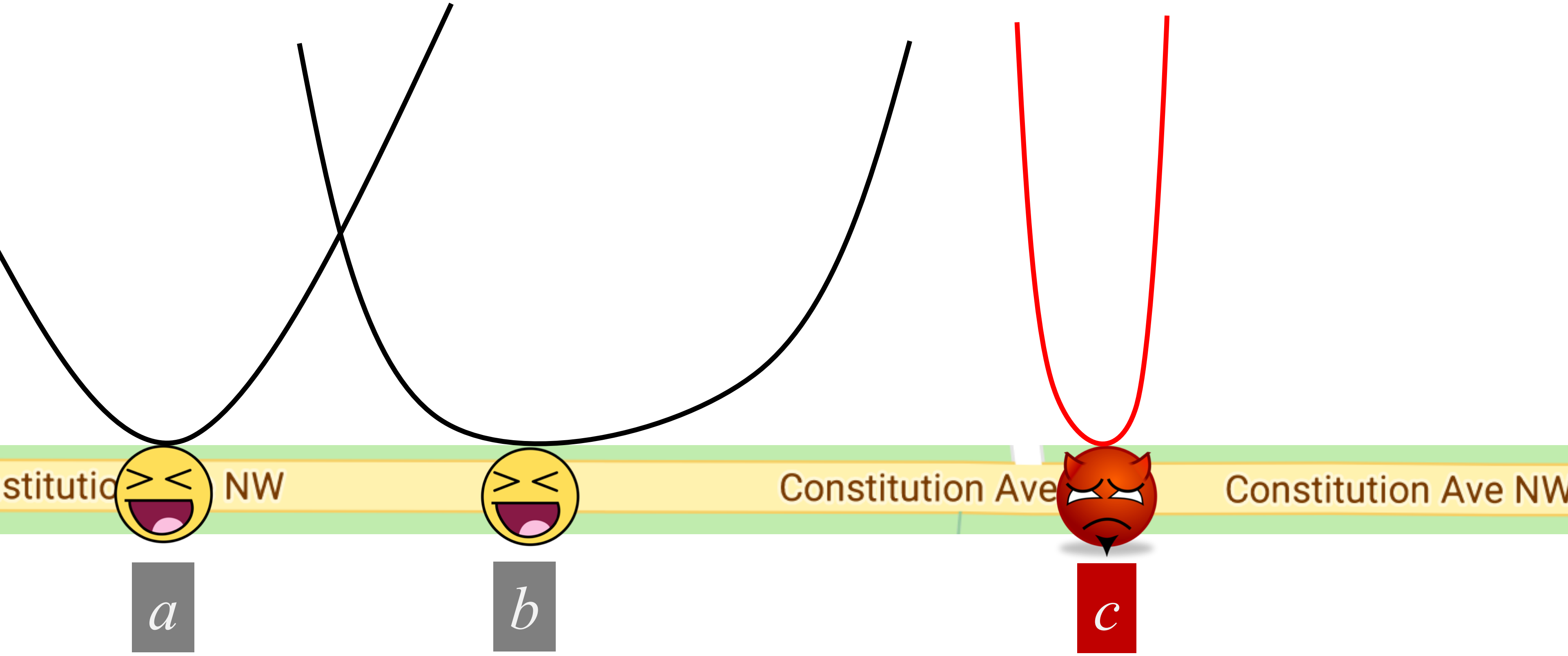
Is this achievable?

It Depends

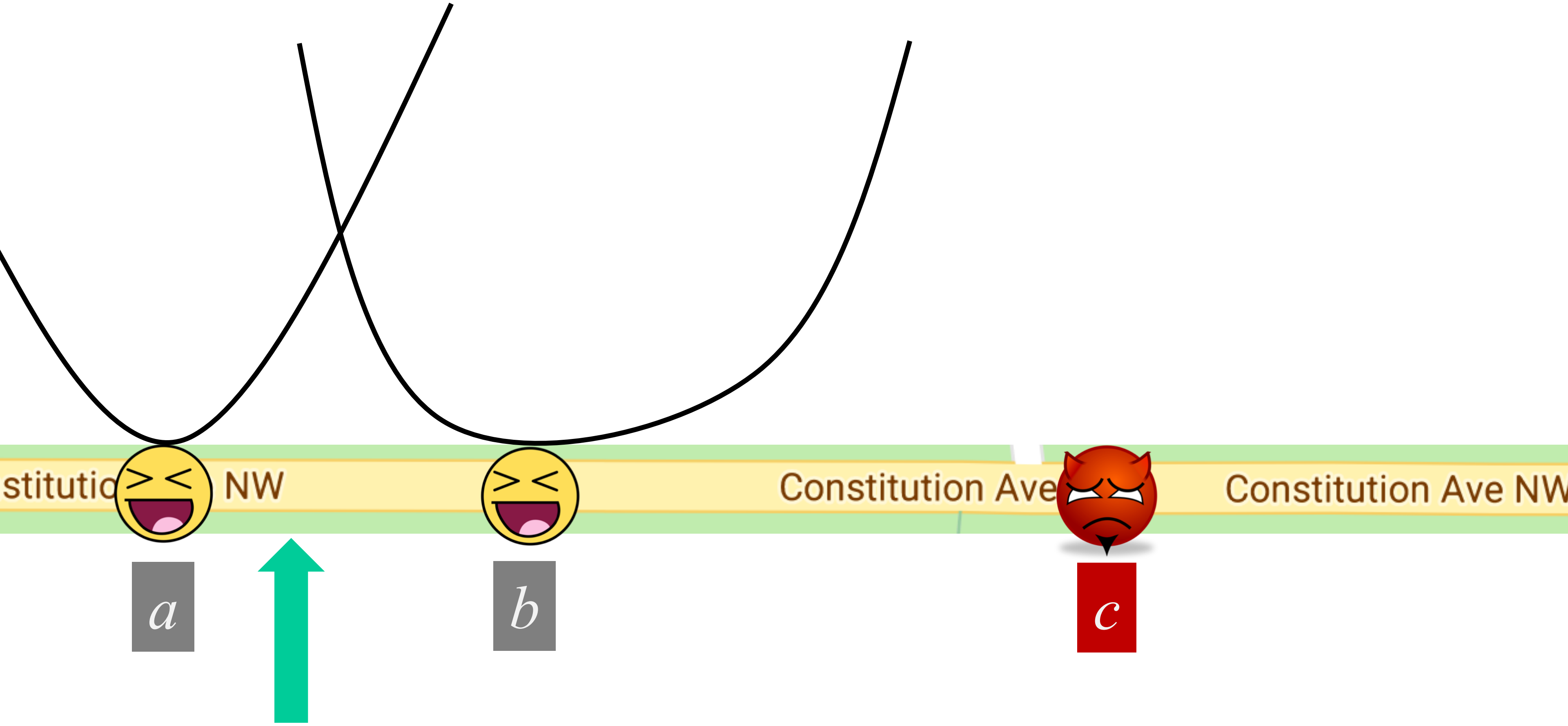
Independent functions

"Enough" redundancy

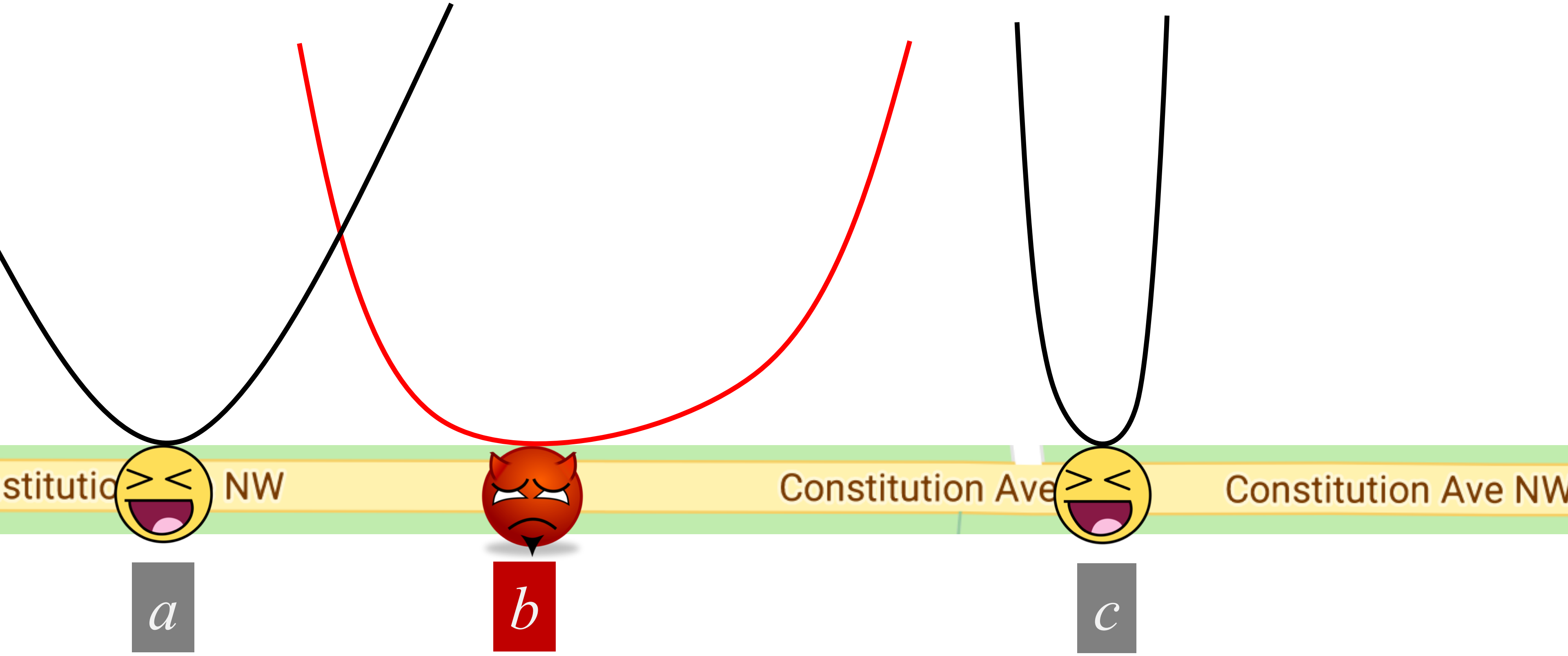
Independent Functions



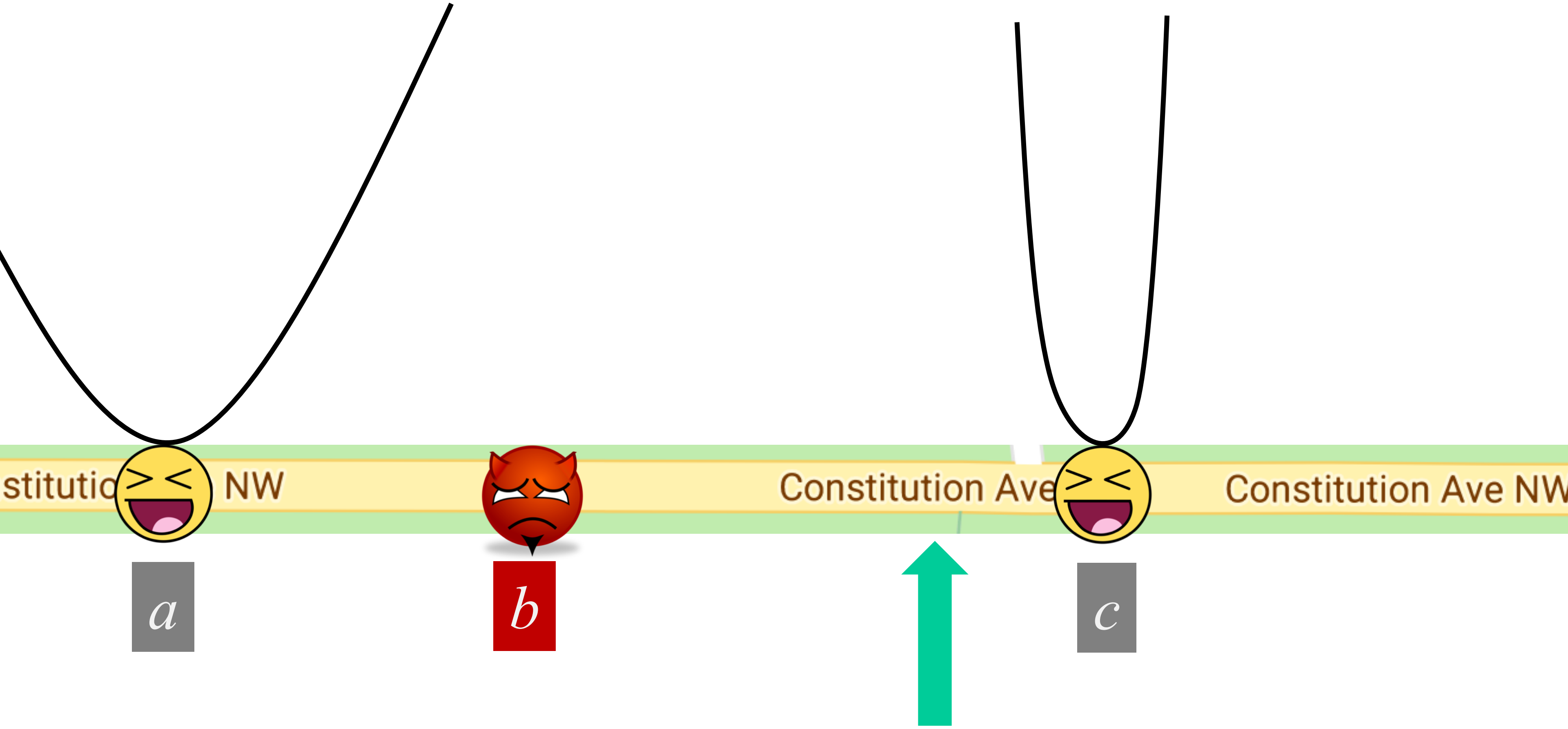
Independent Functions



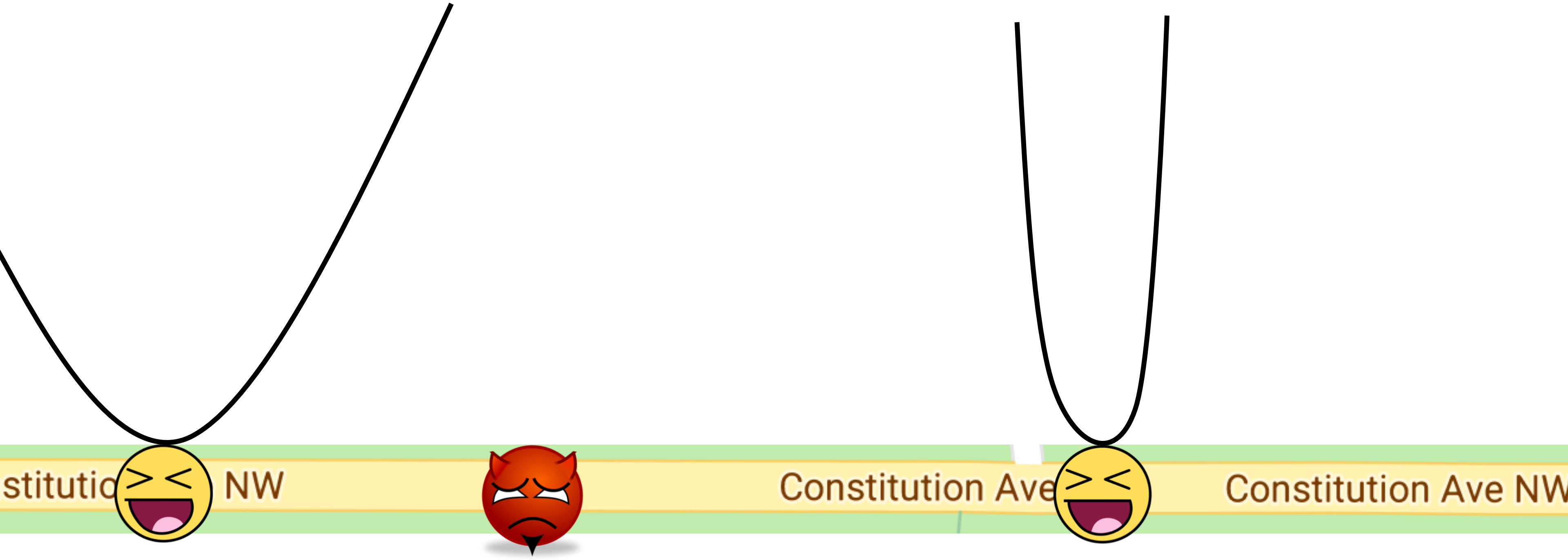
Independent Functions



Independent Functions



Independent Functions



Provably impossible to compute

$$\operatorname{argmin} \sum_{i \in N} f_i(x)$$

$$\operatorname{argmin} \sum_{i \in N} f_i(x)$$

Is this achievable?

Independent functions

"Enough" redundancy

Approximation

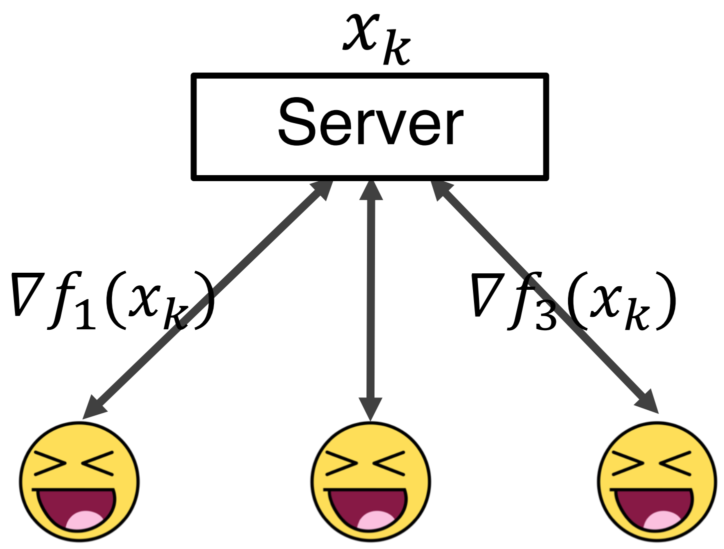
Exact

Parameter Server

- Server maintains estimate x_k

<u>In each iteration</u>

- Agent i
  - Downloads x_k from server
  - Uploads gradient $\nabla f_i(x_k)$



- Server updates estimate

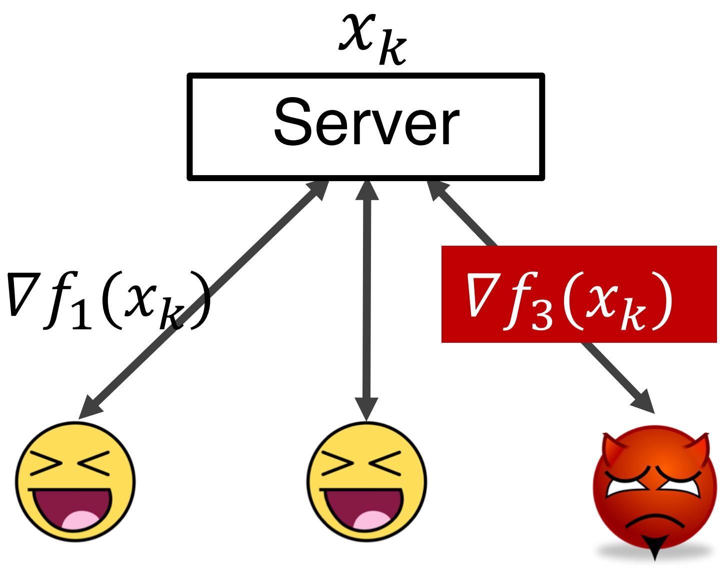
$$x_{k+1} \longleftarrow x_k - \alpha \sum \nabla f_i(x_k)$$

Parameter Server

- Server maintains estimate x_k

<u>In each iteration</u>

- Agent i
  - Downloads x_k from server
  - Uploads gradient $\nabla f_i(x_k)$



- Server updates estimate

$$x_{k+1} \longleftarrow x_k - \alpha \, \text{Filtered}-\text{Gradient}$$

Filter for Scalar Parameter x

t faulty agents

Filter for Scalar Parameter x

t faulty agents

- Discard smallest t and largest t gradients
- Average the rest

Filter for Scalar Parameter x

t faulty agents

- Discard smallest t and largest t gradients
- Average the rest

- t = 1 Filter(1, 4, 6, 0, 2) $= \frac{1+2+4}{3}$

Filter for Scalar Parameter x

t faulty agents

- Discard smallest t and largest t gradients
- Average the rest

- t = 1 $\text{Filter}(1, 4, 6, 0, 2) = \frac{1+2+4}{3}$

What does this achieve?

Our Ideal Goal

Optimize $\sum_{i \in N} f_i(x)$

- Equal importance to each good agent's cost
- Each f_i has identical weight in the aggregate cost

Independent Functions
How good an approximation?

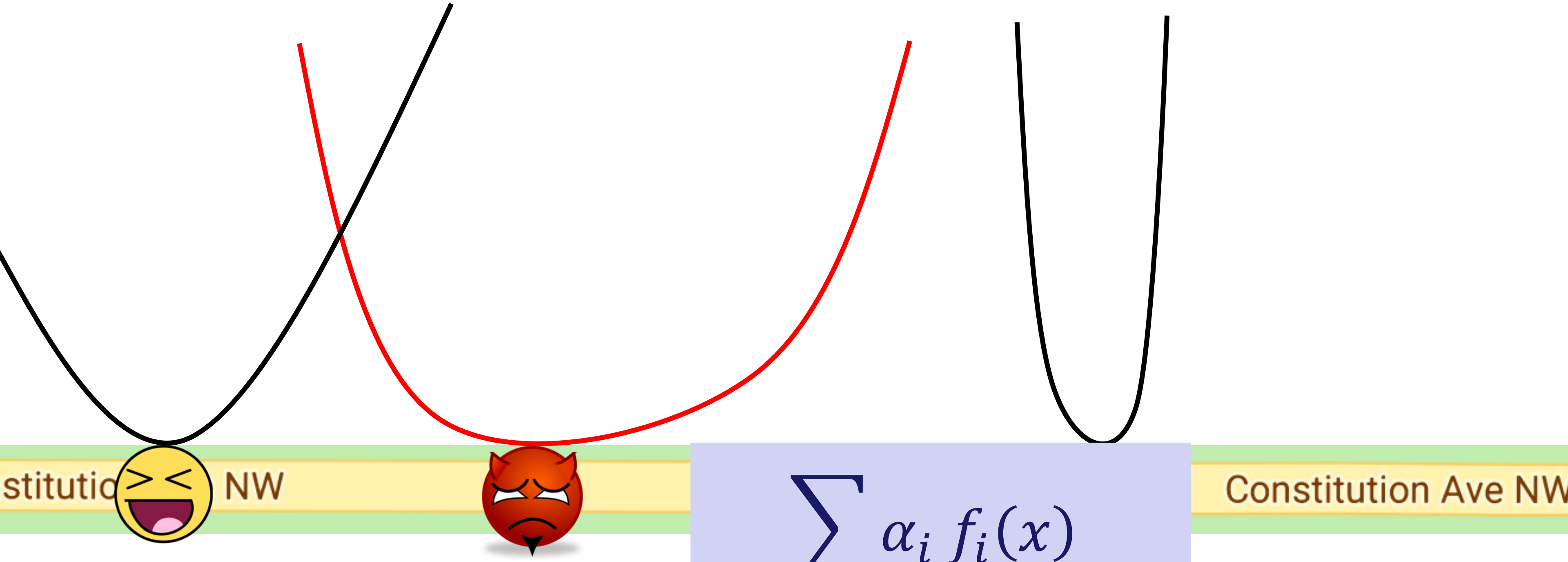
- Instead of uniform weights

$$\sum_{i \in N} f_i(x)$$

the filter achieves unequal weights

$$\sum_{i \in N} \alpha_i \, f_i(x)$$

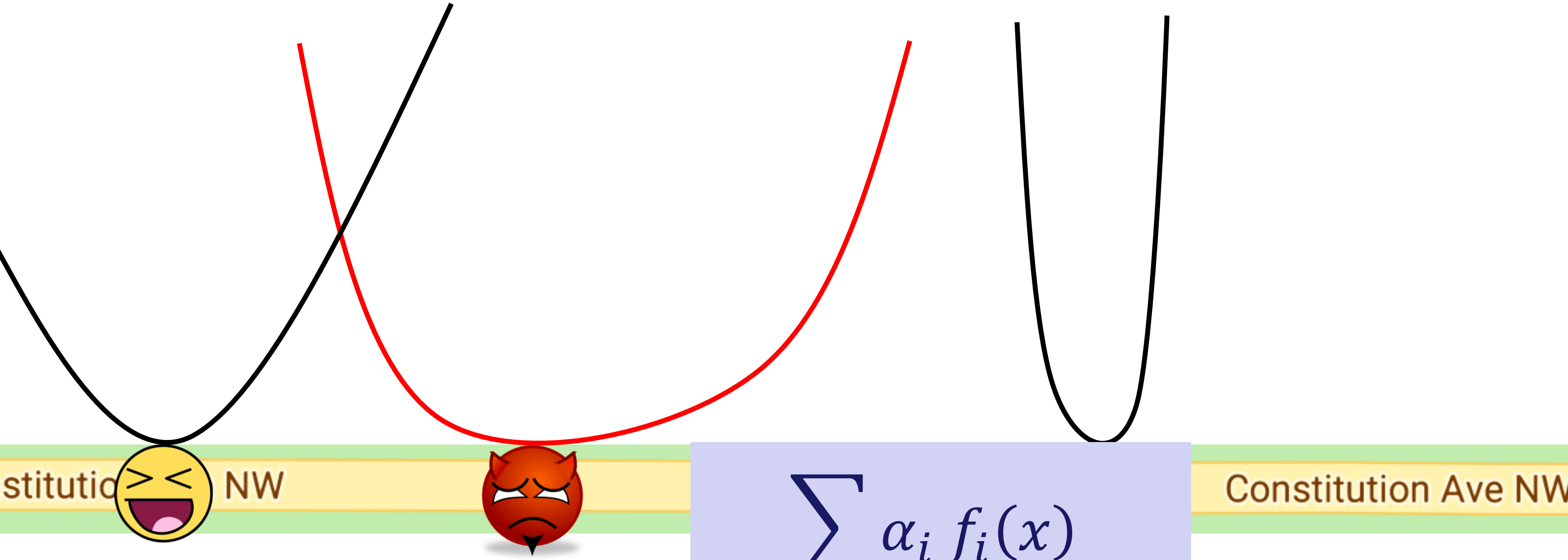
Independent Functions



$$\sum_{i \in N} \alpha_i \, f_i(x)$$

Cost functions of faulty nodes “filtered away”

Independent Functions

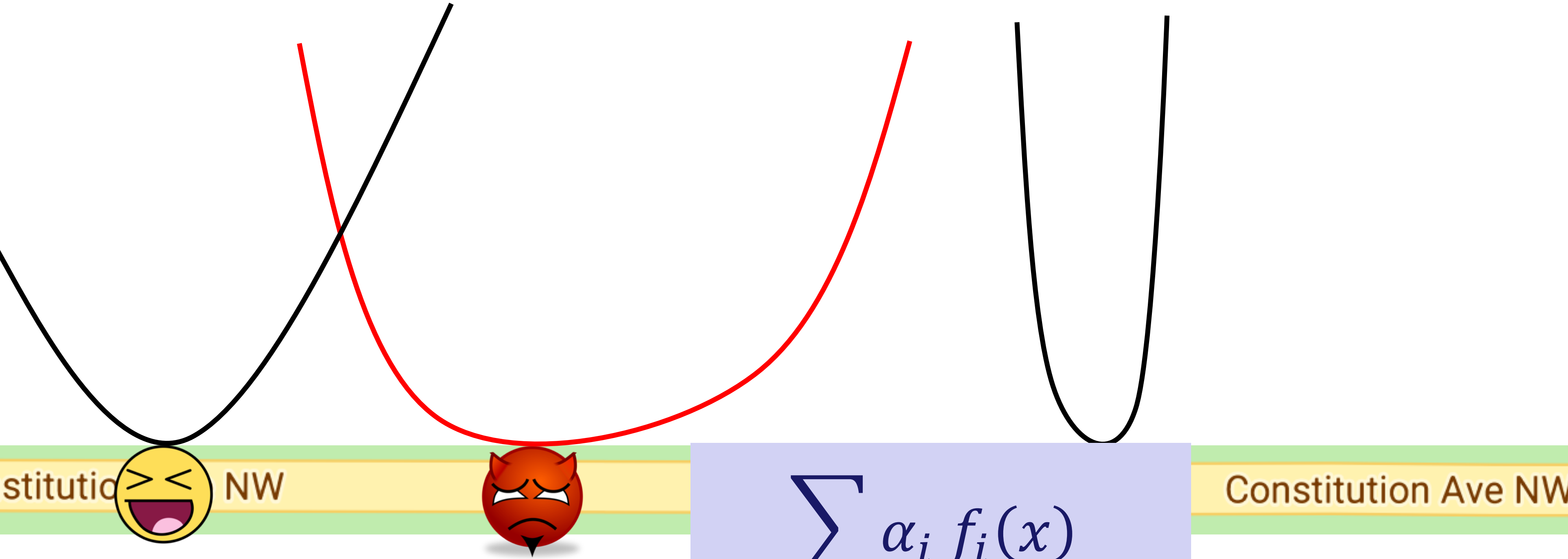


$$\sum_{i \in N} \alpha_i \, f_i(x)$$

Cost functions of faulty nodes “filtered away”

At most t good cost functions also filtered away

Independent Functions



$$\sum_{i \in N} \alpha_i \, f_i(x)$$

Cost functions of faulty nodes “filtered away”

At most t good cost functions also filtered away

Nearly uniform importance to the remaining costs

Independent Functions

How good an approximation?

$$\sum_{i \in N} \alpha_i \, f_i(x)$$

	😆	😆	😆	😆	😆	😆	👿	👿
α	0	0	¼	¼	¼	¼	0	0

Independent Functions

How good an approximation?

$$\sum_{i \in N} \alpha_i \, f_i(x)$$

α	0	0	¼	¼	¼	¼	0	0
α	⅛	⅛	⅛	⅛	¼	¼	0	0

Independent Functions

- Necessary condition in general

$n > (d+1)t$ for d-dimensional parameter x

Bad news for large d

Good News

Cost functions often naturally redundant

Good News
Cost functions often naturally redundant

- Data sets at different agents may be drawn from same distribution

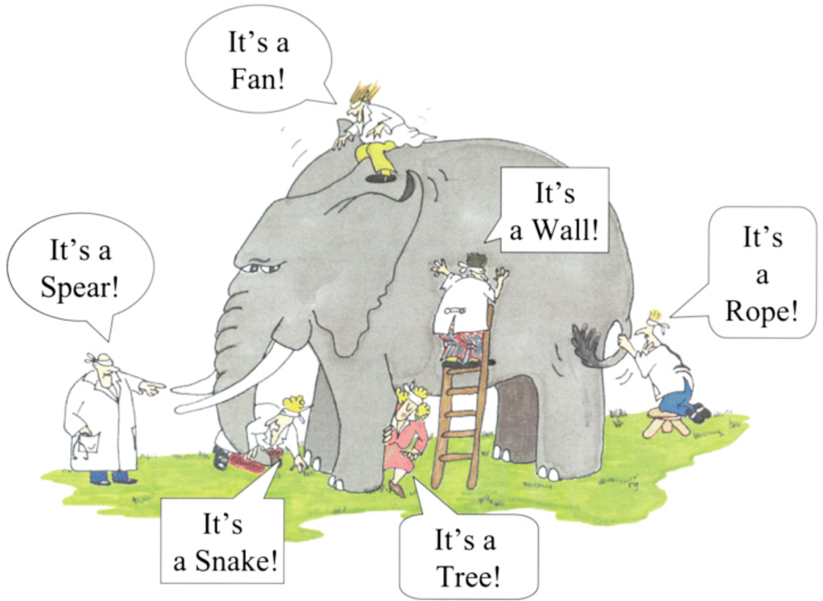
    ➔ In expectation, all cost functions are identical

Good News
Cost functions often naturally redundant

- Data sets at different agents may be drawn from same distribution

  ➔ In expectation, all cost functions are identical

- Observations by different agents conditioned on the same ground truth



Good News

Cost functions often naturally redundant

- With redundancy in functions, possible to compute

$$\operatorname{argmin} \sum_{i \in N} f_i(x)$$

with far fewer than $(d+1)t$ agents

Example of Redundancy

- 2t-redundancy (any dimension d)

  Aggregate cost functions of any $n - 2t$ agents have identical minimum

Example of Redundancy

- 2t-redundancy (any dimension d)

  Aggregate cost functions of any $n - 2t$ agents have identical minimum

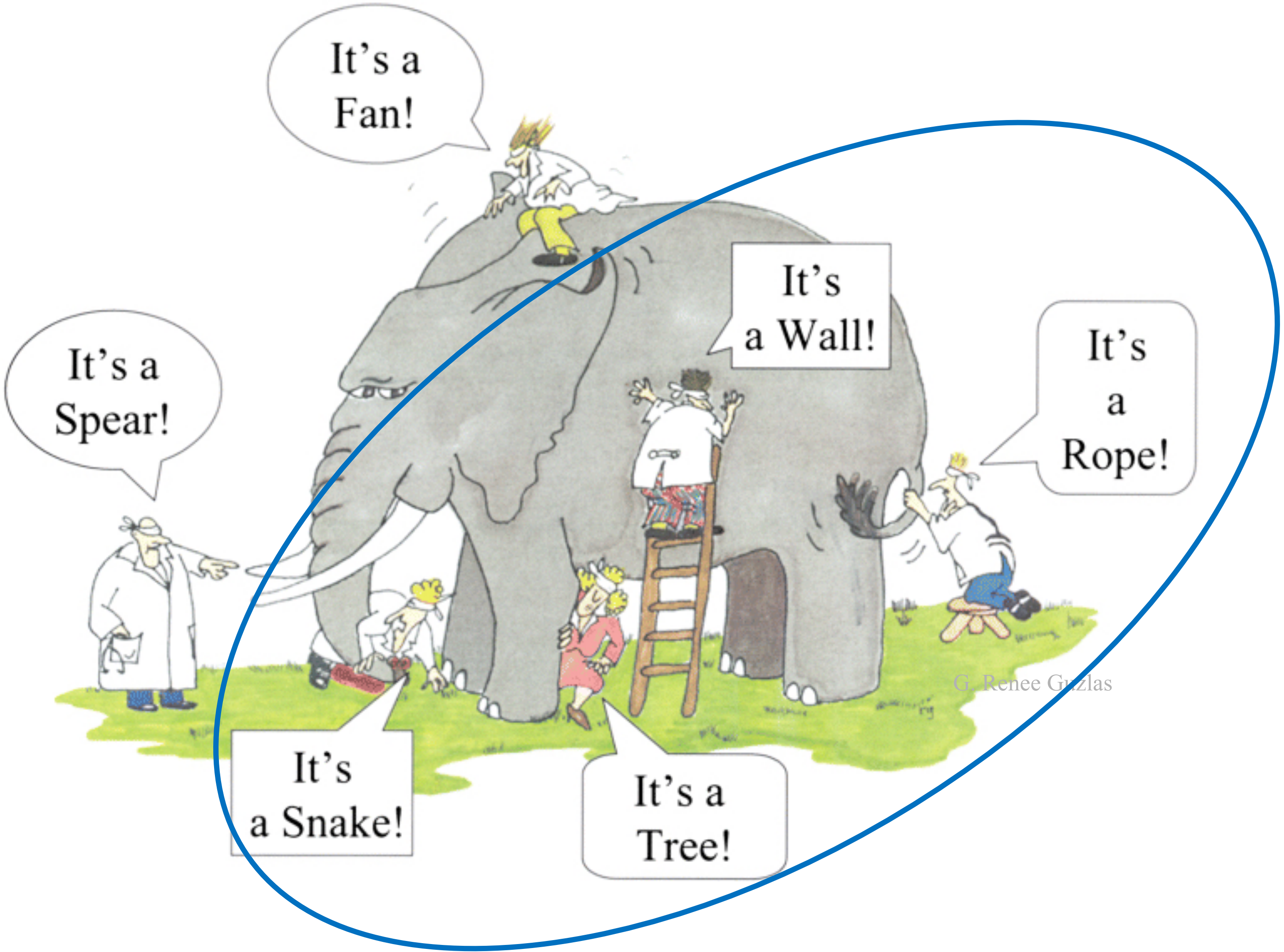
- There exists x^* such that for any set S of $n - 2t$ agents,

$$x^* = \operatorname{argmin} \sum_{i \in S} f_i(x)$$

2t-redundancy

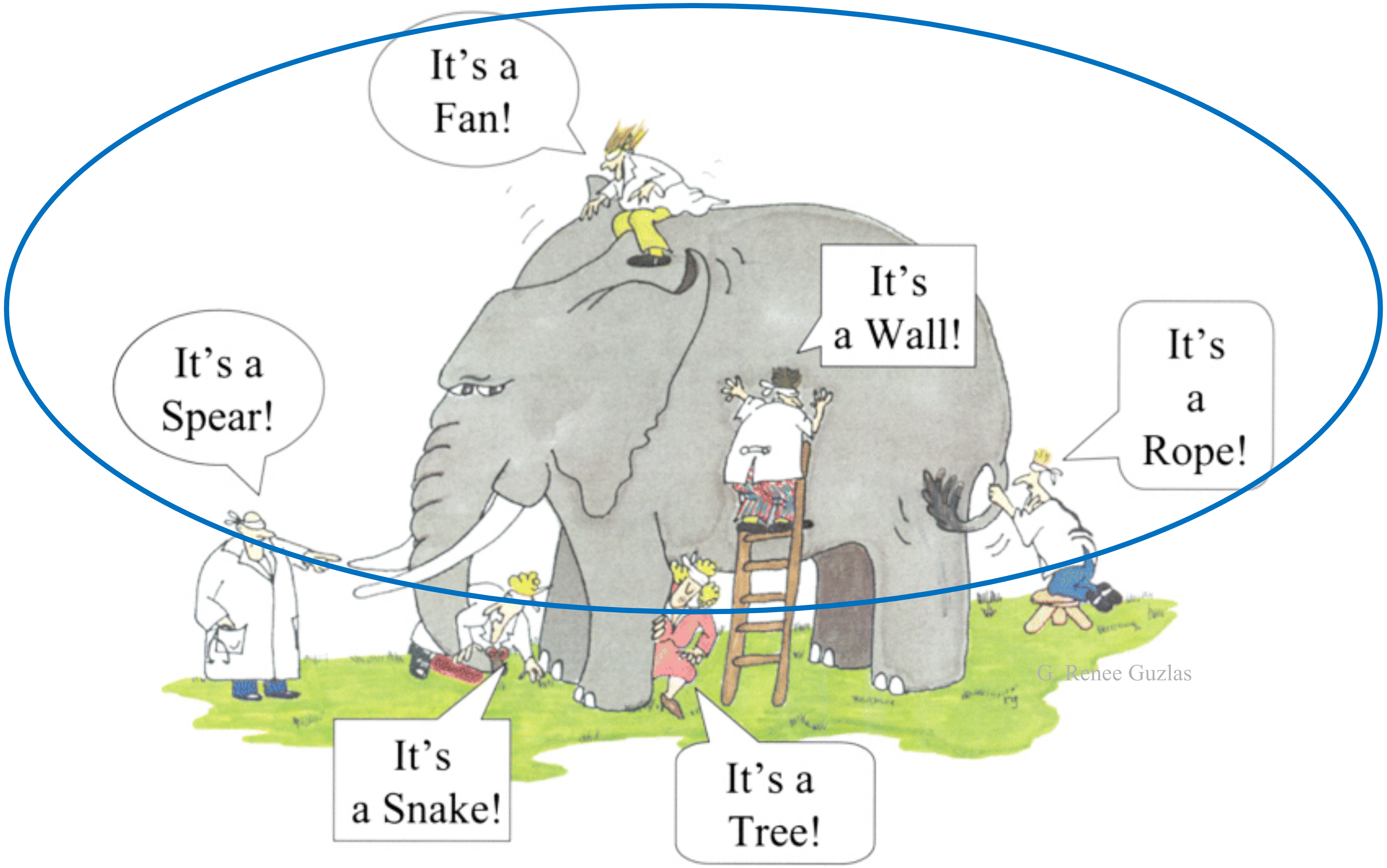
$n = 6$

$t = 1$



2t-redundancy

$n = 6$
$t = 1$

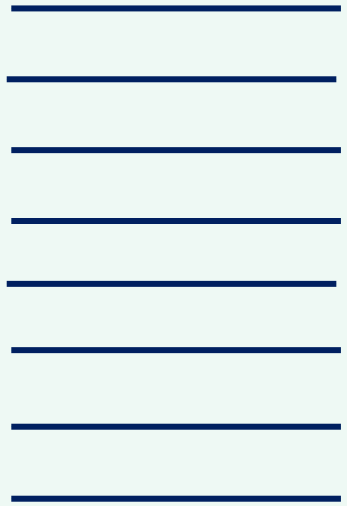


Linear Regression

Linear Regression

- d-dimensional x^*

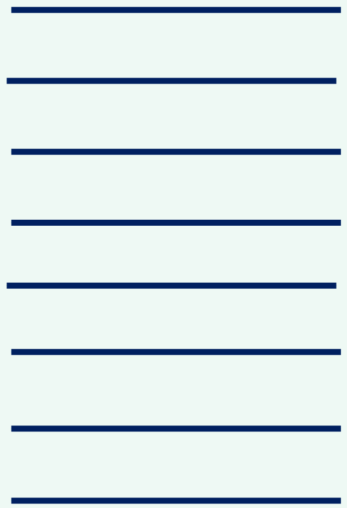
- n-by-d matrix A



Linear Regression

- d-dimensional x^*

- n-by-d matrix A

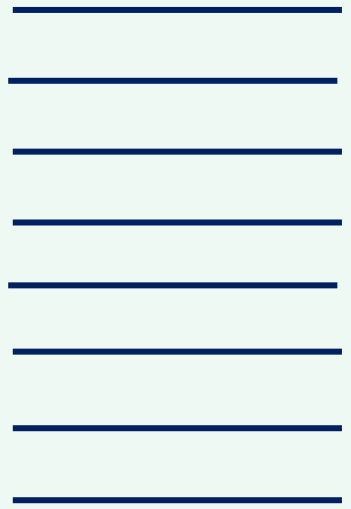


- n-dimensional y

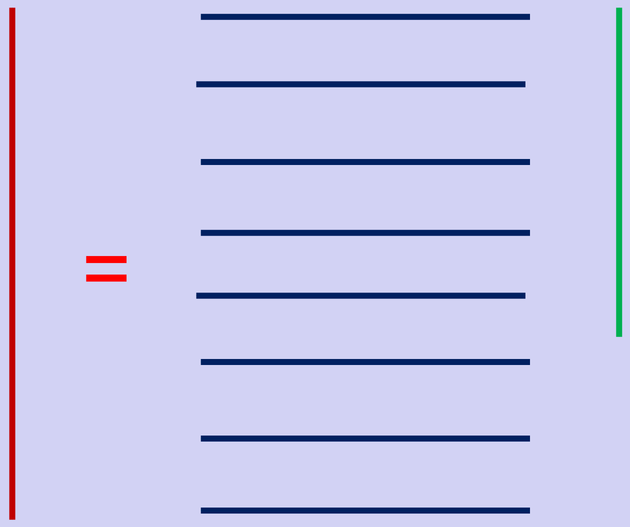
Linear Regression

- d-dimensional x^*

- n-by-d matrix A



- n-dimensional y



$$y = A\,x^*$$

Distributed Linear Regression

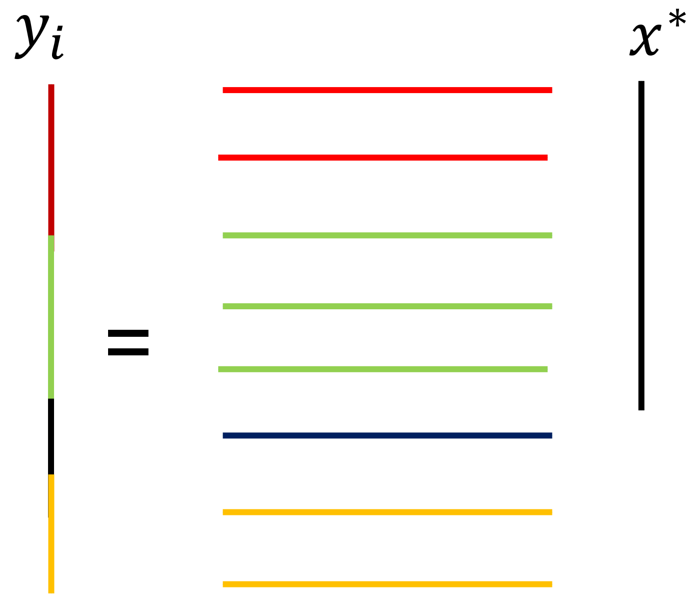
- Agent i knows A_i and y_i where

$$y_i = A_i\, x^*$$

Distributed Linear Regression

- Agent i knows A_i and y_i where

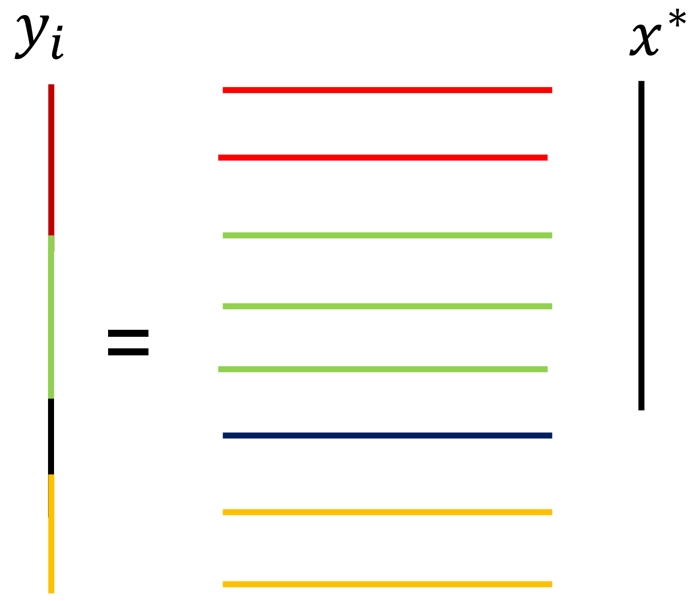
$$y_i = A_i x^*$$



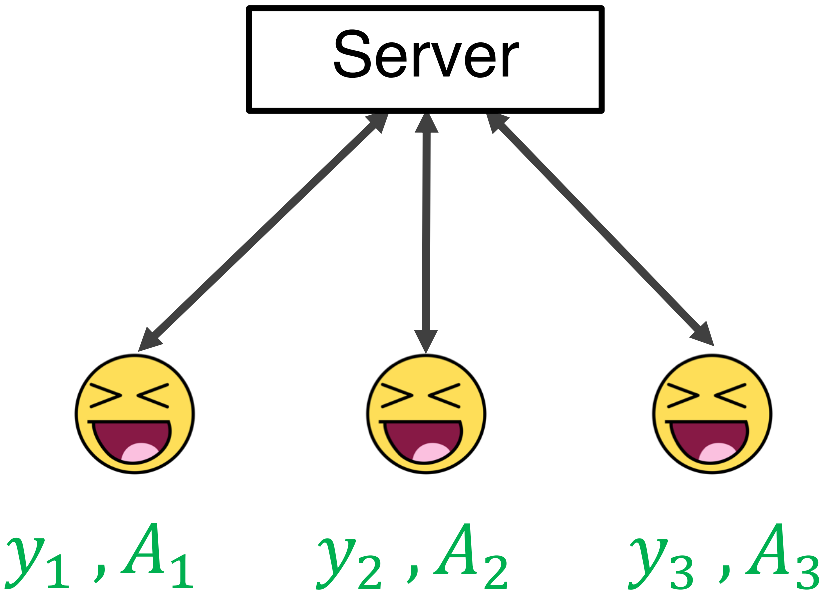
Distributed Linear Regression

- Agent i knows A_i and y_i where

$$y_i = A_i\, x^*$$



- Determine x^* allowing for t agents to be faulty


Server
y_1 , A_1
y_2 , A_2
y_3 , A_3

2t-Redundancy … Linear Regression

- Any $n - 2t$ agents have enough information to compute x^*

2t-Redundancy … Linear Regression

- Any $n - 2t$ agents have enough information to compute x^*

- Define cost function of agent i as

$$f_i(x) = (y_i - A_i x)^2$$

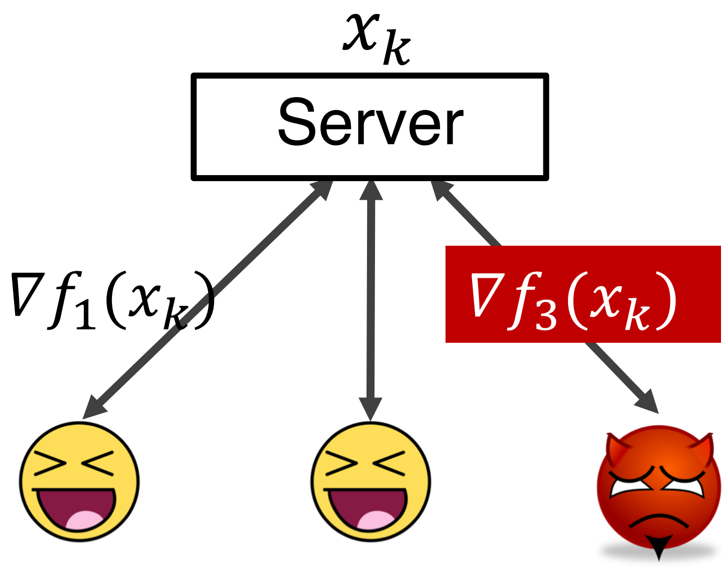
- The cost functions satisfy $2t$-redundancy

Parameter Server

- Server maintains estimate x_k

<u>In each iteration</u>

- Agent i
  - Downloads x_k from server
  - Uploads gradient $\nabla f_i(x_k)$



- Server updates estimate

$$x_{k+1} \longleftarrow x_k - \alpha \, \text{Filtered−Gradient}$$

2t-Redundancy … Linear Regression

- Simple gradient filter based on gradient norms

Clip the largest t norms to equal $t+1^{\text{th}}$ norm

2t-Redundancy … Linear Regression

- Simple gradient filter based on gradient norms

Clip the largest t norms to equal $t + 1$th norm

- Distributed algorithm computes x^* such that $y_i = A_i x^*$ for each good agent i

$$\operatorname{argmin} \sum_{i \in N} f_i(x)$$

Is this achievable?

Independent functions

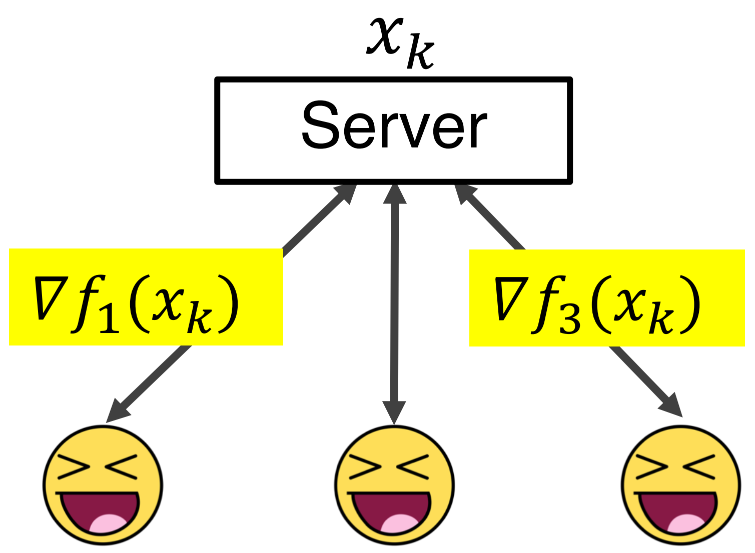
“Enough” Redundancy

Approximation

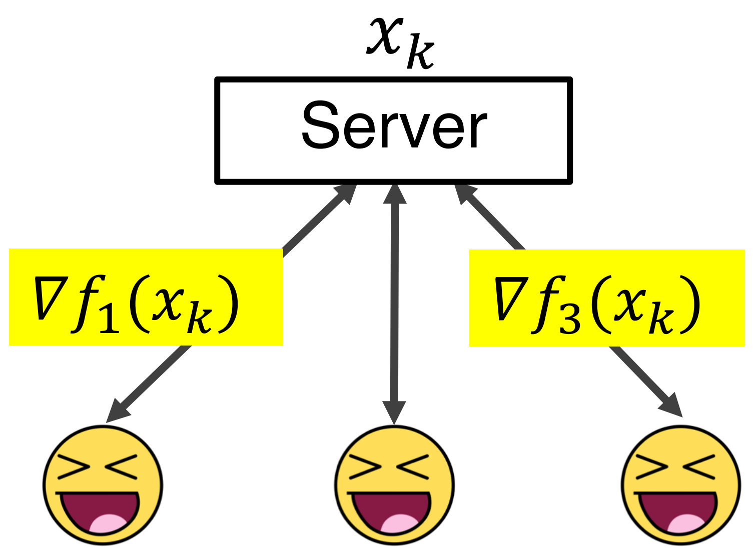
Exact

Privacy-Preserving Optimization

Communication Leaks Information



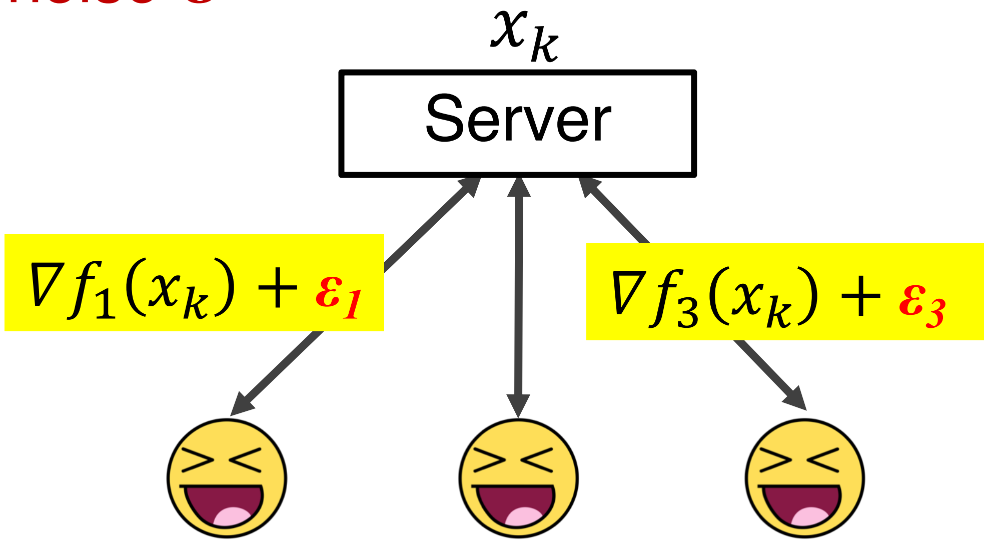
Communication Leaks Information



Server can use gradients to infer polynomial cost functions (up to a constant)

Privacy Techniques

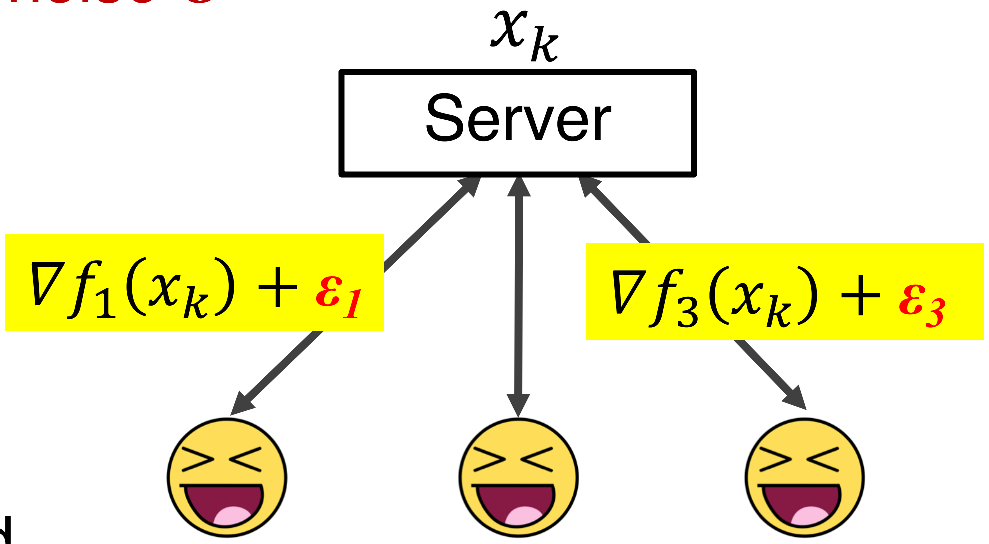
Differential privacy … add noise ε



x_k
Server
$\nabla f_1(x_k) + \varepsilon_1$
$\nabla f_3(x_k) + \varepsilon_3$

Privacy Techniques

Differential privacy … add noise ε



x_k
Server
$\nabla f_1(x_k) + \varepsilon_1$
$\nabla f_3(x_k) + \varepsilon_3$

- Optimality compromised due to the noise

$$\neq \operatorname{argmin} \sum f_i(x)$$

Privacy Techniques

Homomorphic encryption

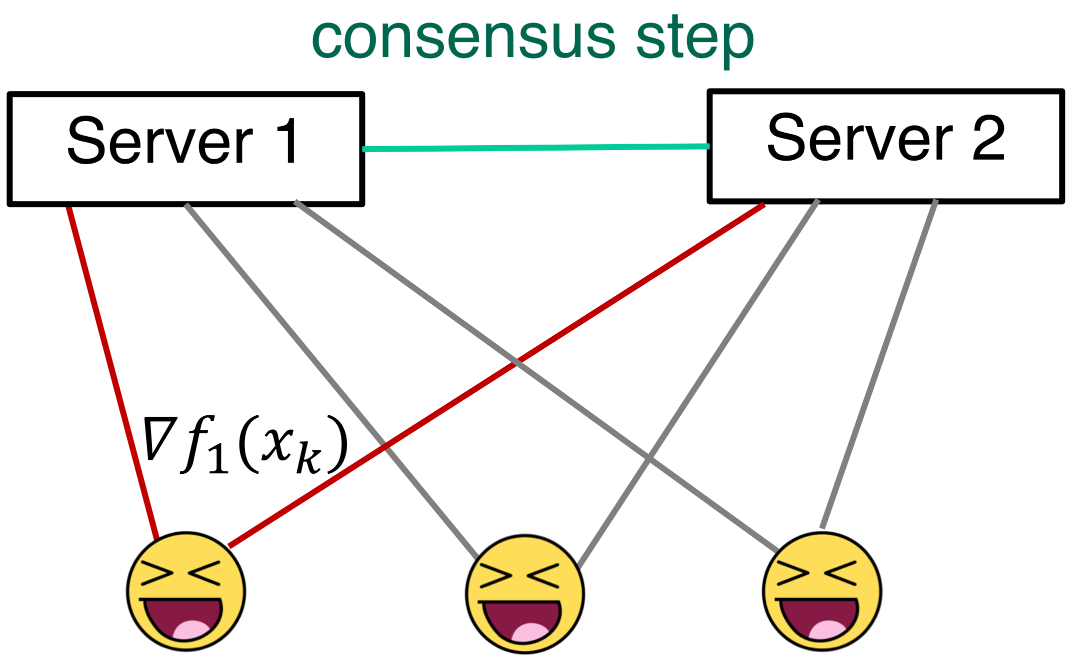
- Expensive

Our Approach

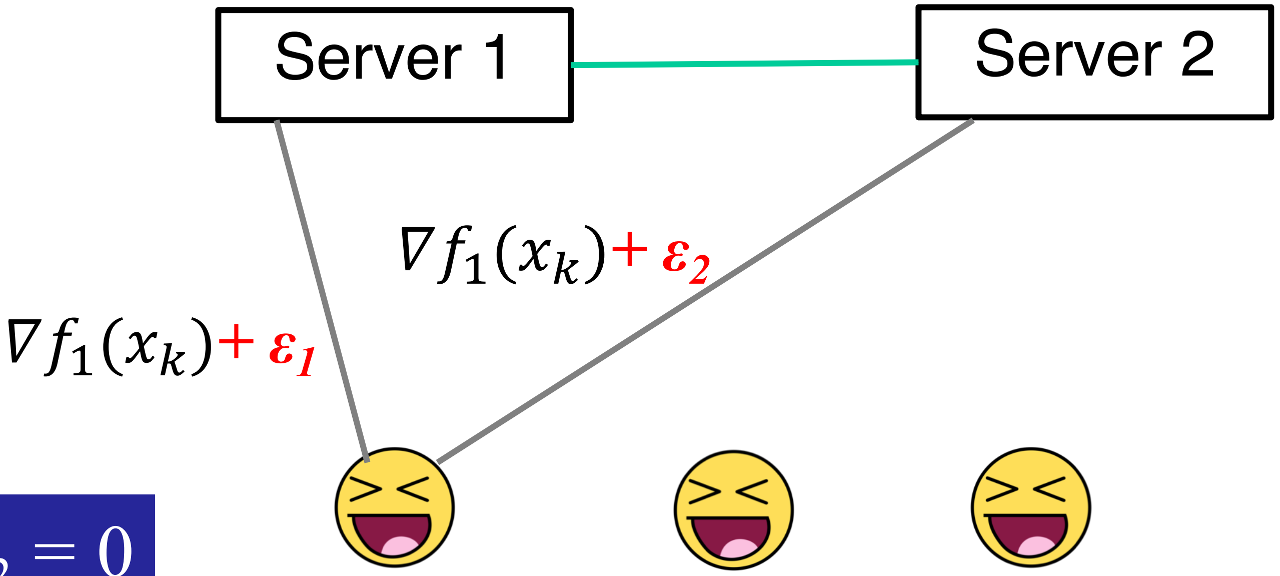
- Motivated by secret sharing & differential privacy

➔ Add cancellable noise

Multiple Parameter Servers

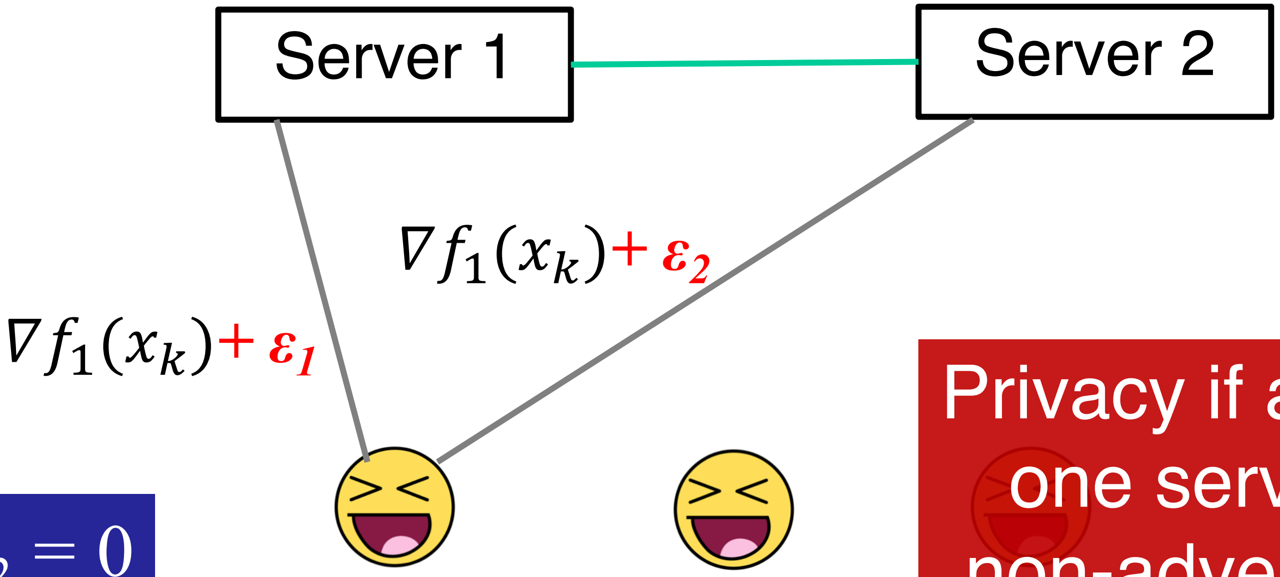


Privacy


Server 1
Server 2
$\nabla f_1(x_k) + \varepsilon_2$
$\nabla f_1(x_k) + \varepsilon_1$

$\varepsilon_1 + \varepsilon_2 = 0$
over time

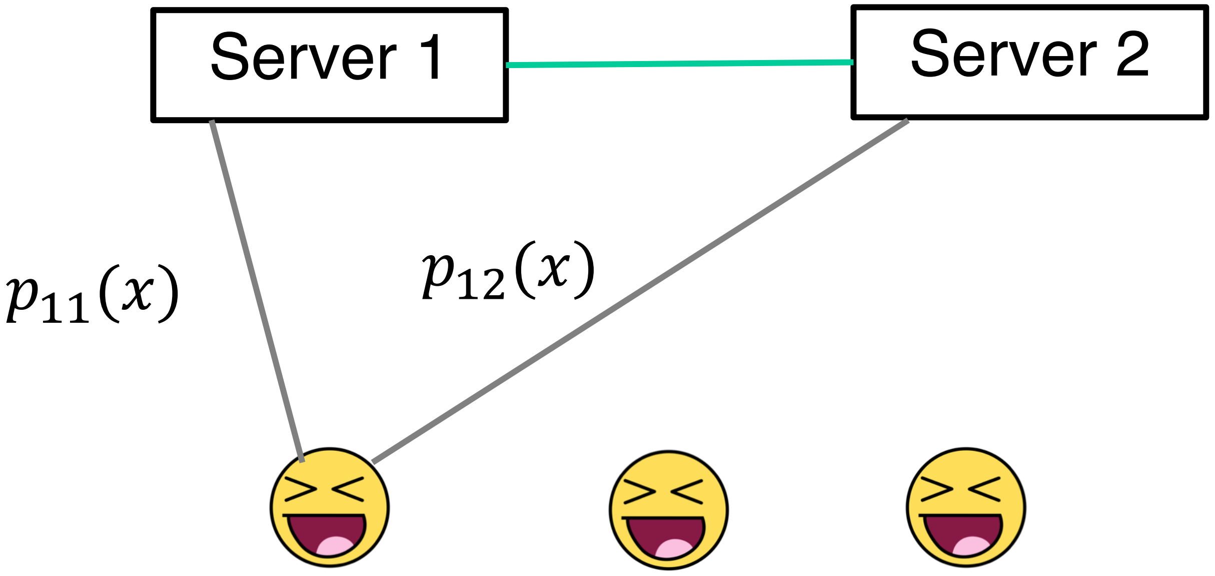
Privacy


Server 1
Server 2
$\nabla f_1(x_k) + \varepsilon_2$
$\nabla f_1(x_k) + \varepsilon_1$

$\varepsilon_1 + \varepsilon_2 = 0$
over time

Privacy if at least one server is non-adversarial

Privacy

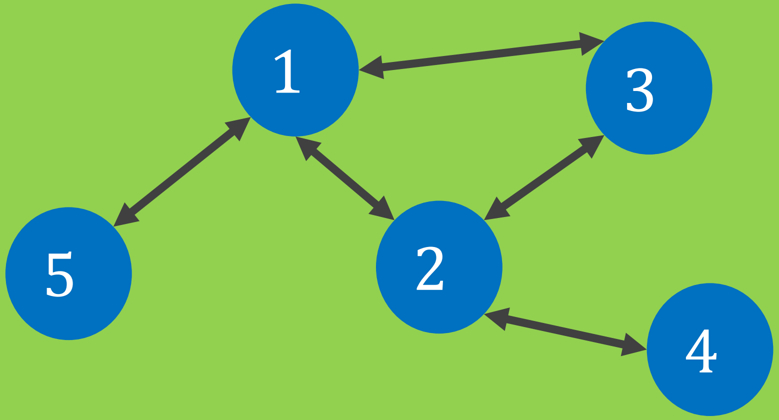


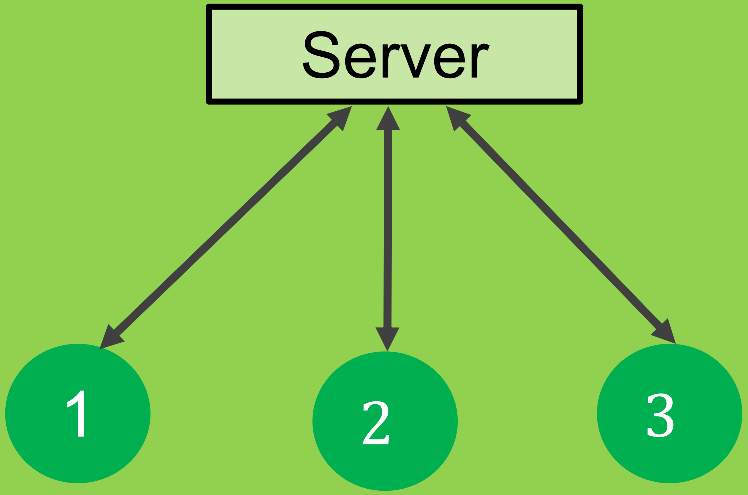
$$p_{11}(x) + p_{12}(x) = f_1(x)$$

Summary

argmin $\sum f_i(x)$

- Fault-tolerance … gradient filters
- Privacy … cancellable noise





Moral of the Story

Decentralized Control/Optimization

Distributed Computing

Acknowledgements

Shripad Gade

Lili Su

Nirupam Gupta

Thanks!

disc.georgetown.domains

Thanks!

disc.georgetown.domains

Averaging ➔ Optimization

- Input of agent i = a_i

- Average of $a_i's$

Averaging ➔ Optimization

- Input of agent i = a_i

- Average of $a_i's$ = $\text{argmin} \sum f_i(x)$

where $f_i(x) = (x - a_i)^2$

Hajnal 1958
Weak ergodicity of nonhomogeneous Markov chains

Distributed Computing

DeGroot 1974
Reaching a consensus

1980: Pease, Shostak, Lamport
Byzantine consensus

1983: Fischer, Lynch, Paterson
Asynchronous consensus impossibility result

Decentralized Control

Tsitsiklis 1984

1986: Dolev et al.
Approximate Byzantine consensus

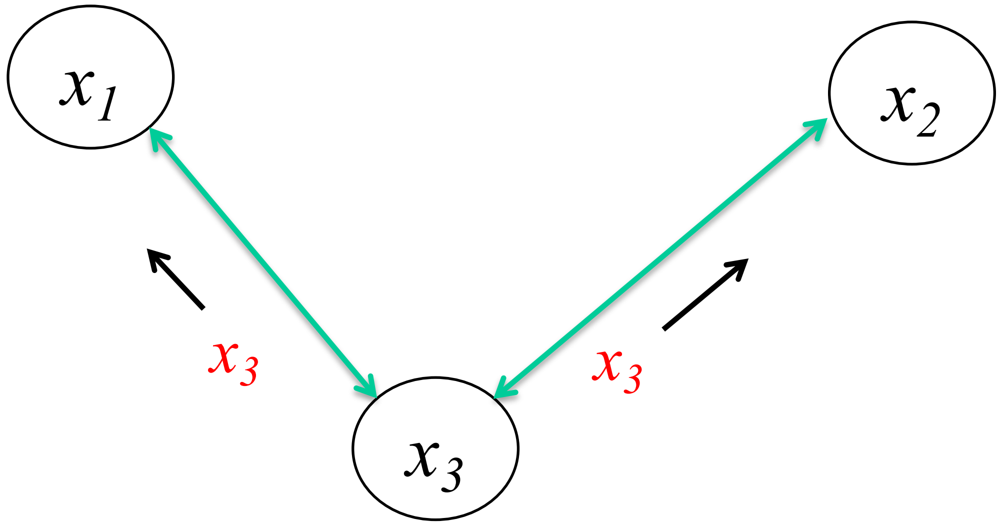
Jadbabaei, Lin, Morse 2003
Flocking problem

Nedich, Ozdaglar 2009

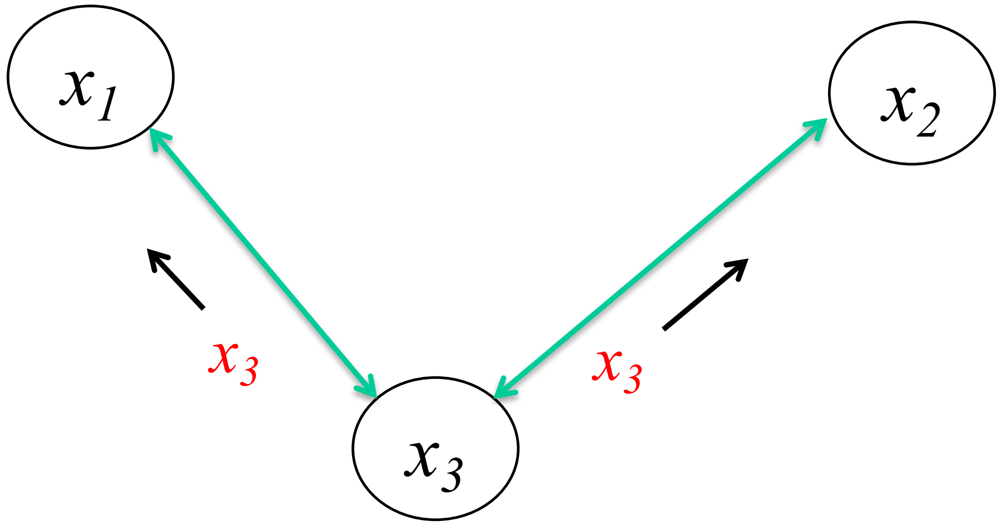
Distributed Optimization

$$f(x) = \sum f_i(x)$$

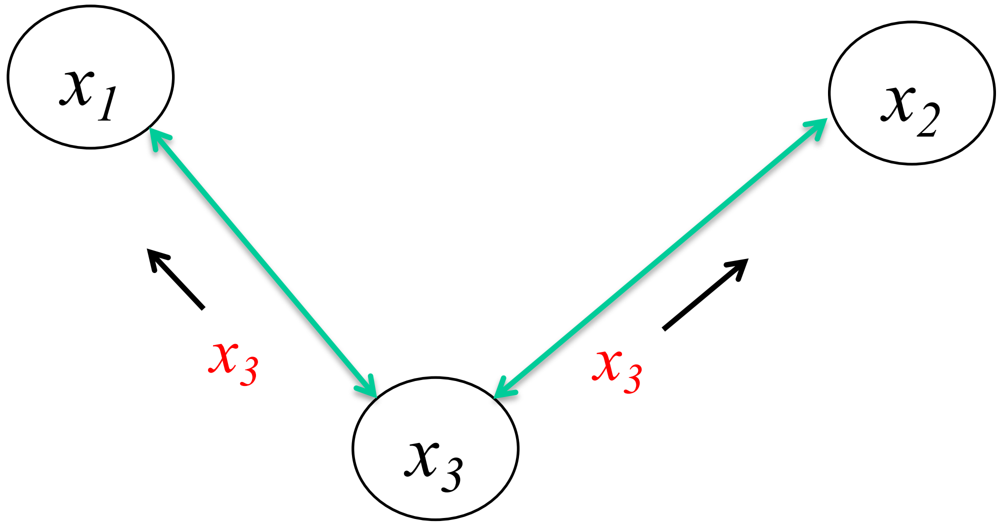
- Each agent maintains an estimate
- Local estimates shared with neighbors & updated
- Estimates converge to optimum



Example based on [Nedic and Ozdaglar, 2009]



$$x_1 \longleftarrow \frac{2}{3}x_1 + \frac{1}{3}x_3 - \alpha \nabla f_1(x_1)$$



$$x_1 \longleftarrow \frac{2}{3}x_1 + \frac{1}{3}x_3 - \alpha \nabla f_1(x_1)$$

$$x_3 \longleftarrow \frac{1}{3}x_1 + \frac{1}{3}x_2 + \frac{1}{3}x_3 - \alpha \nabla f_3(x_3)$$

Distributed Optimization

As time → ∞

- Consensus: All agents converge to same estimate
- Optimality

$$\text{argmin} \sum f_i(x)$$