Scientific Computing in Economics and Finance
Past, Present and Future

John Stachurski

Tokyo College and Australian National University

April 25th 2025

What do economists do?

- Set interest rates at the BOJ
- Assess tax policies
- Advise on competition policy
- Assess pension plans
- Study the impact of COVID / COVID relief policy

Many problems addressed by economists are quantitative

Examples.

- How will a 10% increase in the top income tax rate affect GDP over the next decade?
- How will a two year increase in the retirement age affect government debt?
- What will be the impact of a 25 BPS $\uparrow$ in the federal funds rate on unemployment next year?

Quantitative problems require mathematical/computational modeling

All science requires mathematical modeling

The goal of the scientist is to comprehend the phenomena of the universe that he observes around him.

To prove that he understands he must be able to predict.

To predict quantitatively one must have a mechanism for producing numbers.

This necessarily entails a mathematical model.

– Richard Bellman (1920 – 1984)

All science requires mathematical modeling

The goal of the scientist is to comprehend the phenomena of the universe that he observes around him.

To prove that he understands he must be able to predict.

To predict quantitatively one must have a mechanism for producing numbers.

This necessarily entails a mathematical model.

– Richard Bellman (1920 – 1984)

How should economists build models?

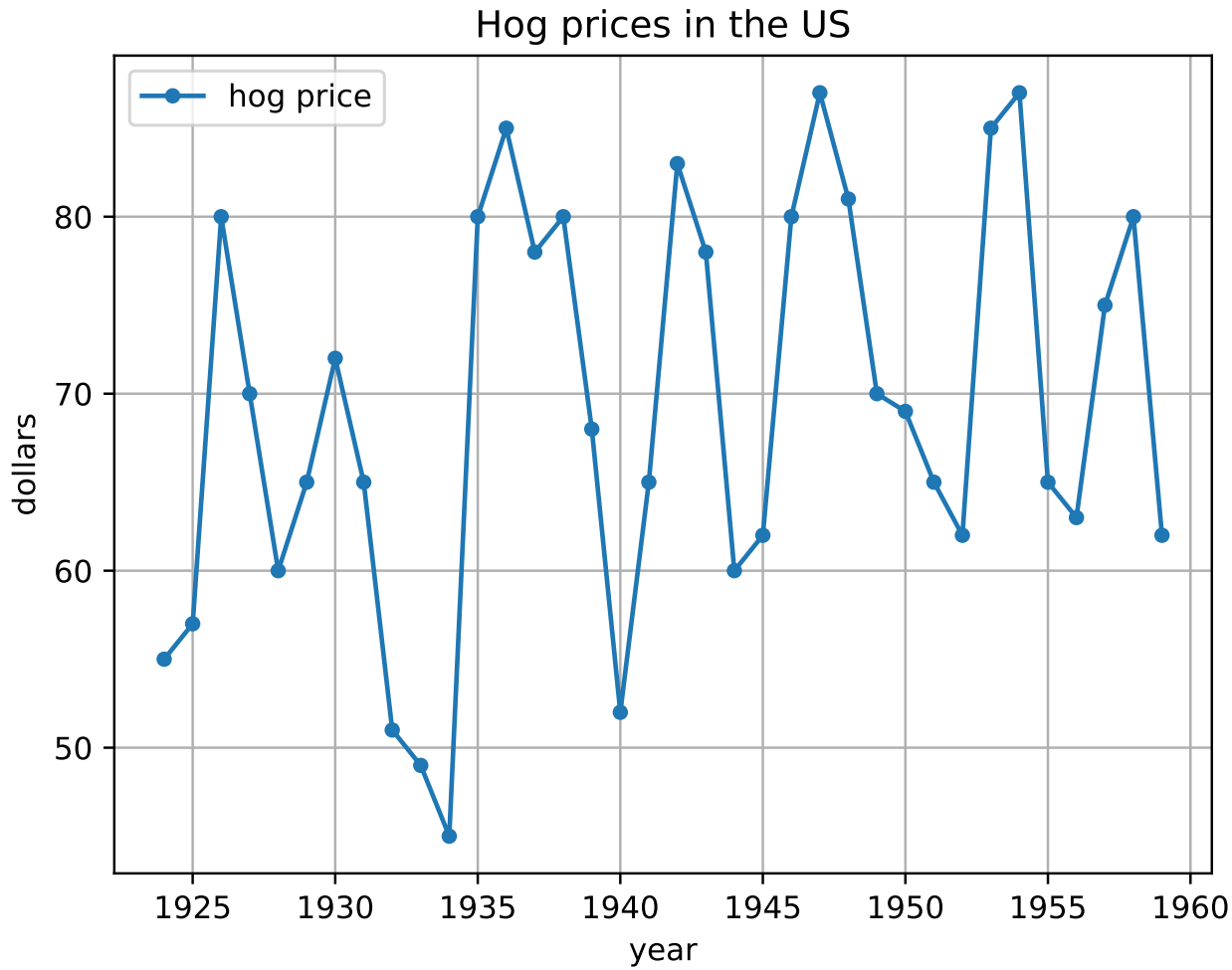
Why is economics different to astrophysics, chemistry, etc.?

- Economic processes are nonstationary (no immutable laws)
- Economic outcomes depend on human choices
- Human choices depend on beliefs, incentives, etc.

Example: the cobweb model

An "old" economic model

- Benner (1876)
- Haas and Ezekil (1926)
- Ricci (1930)
- Kaldor (1934, 1938)
- Ezekil (1938)
- Harlow (1960)
- Rosen, Murphy, and Scheinkman (1994)


Hog prices in the US
hog price
dollars
year
80
70
60
50
1925
1930
1935
1940
1945
1950
1955
1960

Hypotheses:

- Farmers need time to raise hogs (say, one "period")
- Farmers forecast future prices using current prices

Outcomes:

1. Suppose price is currently high
2. Farmers $\uparrow$ capacity, shift towards hog production
3. Next period, high supply floods the market, prices $\downarrow$
4. Seeing this low price, farmers $\downarrow$ capacity
5. Next period, supply is low and prices $\uparrow$ ...

In this scenario,

$$q^d(p_t) = q^s(p^e_{t-1})$$

- p^e_{t-1} is the **expected** time t price, formed at $t-1$

But how to farmers form expectations?

First guess:

$$p^e_{t-1} = p_{t-1}$$

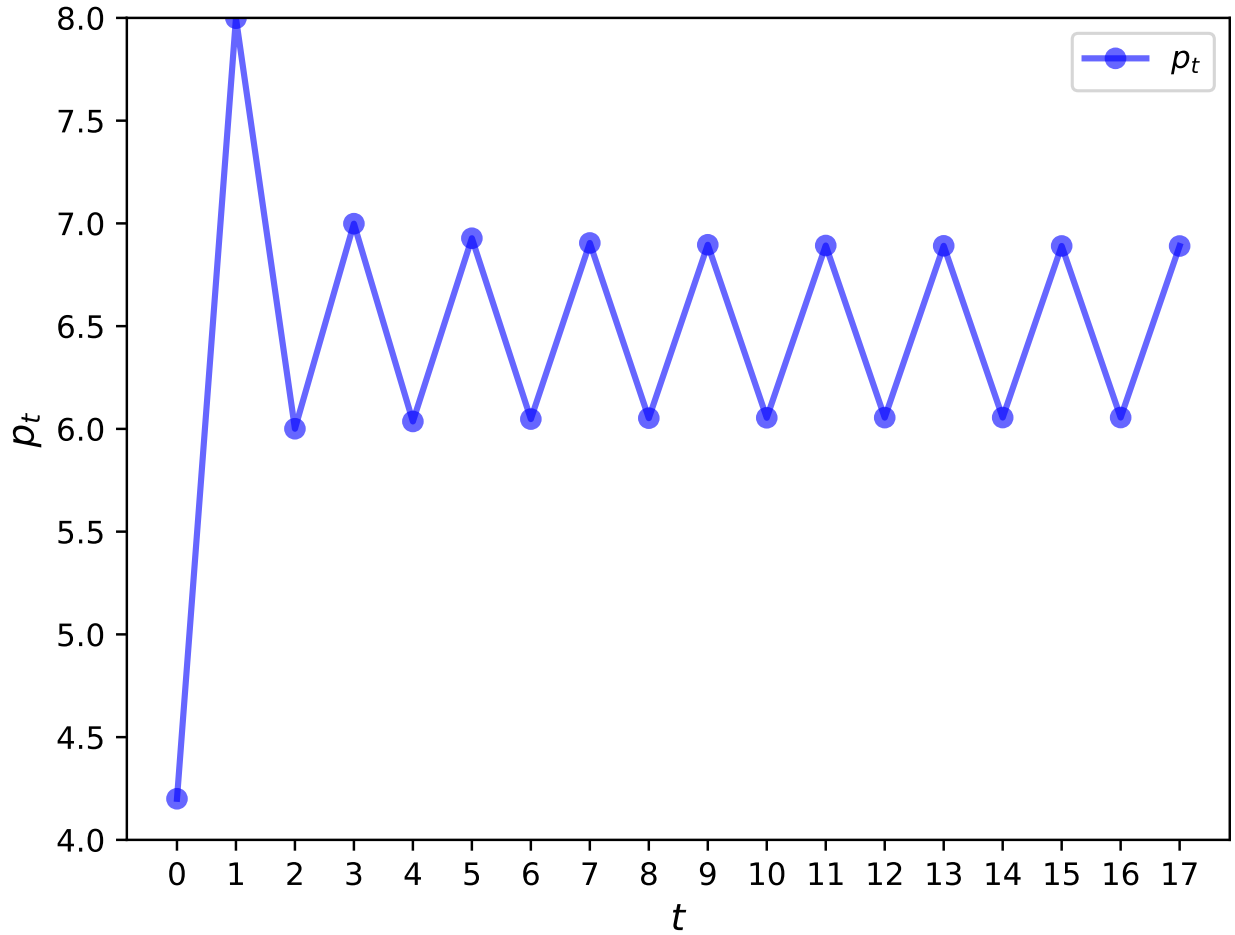
So now we have

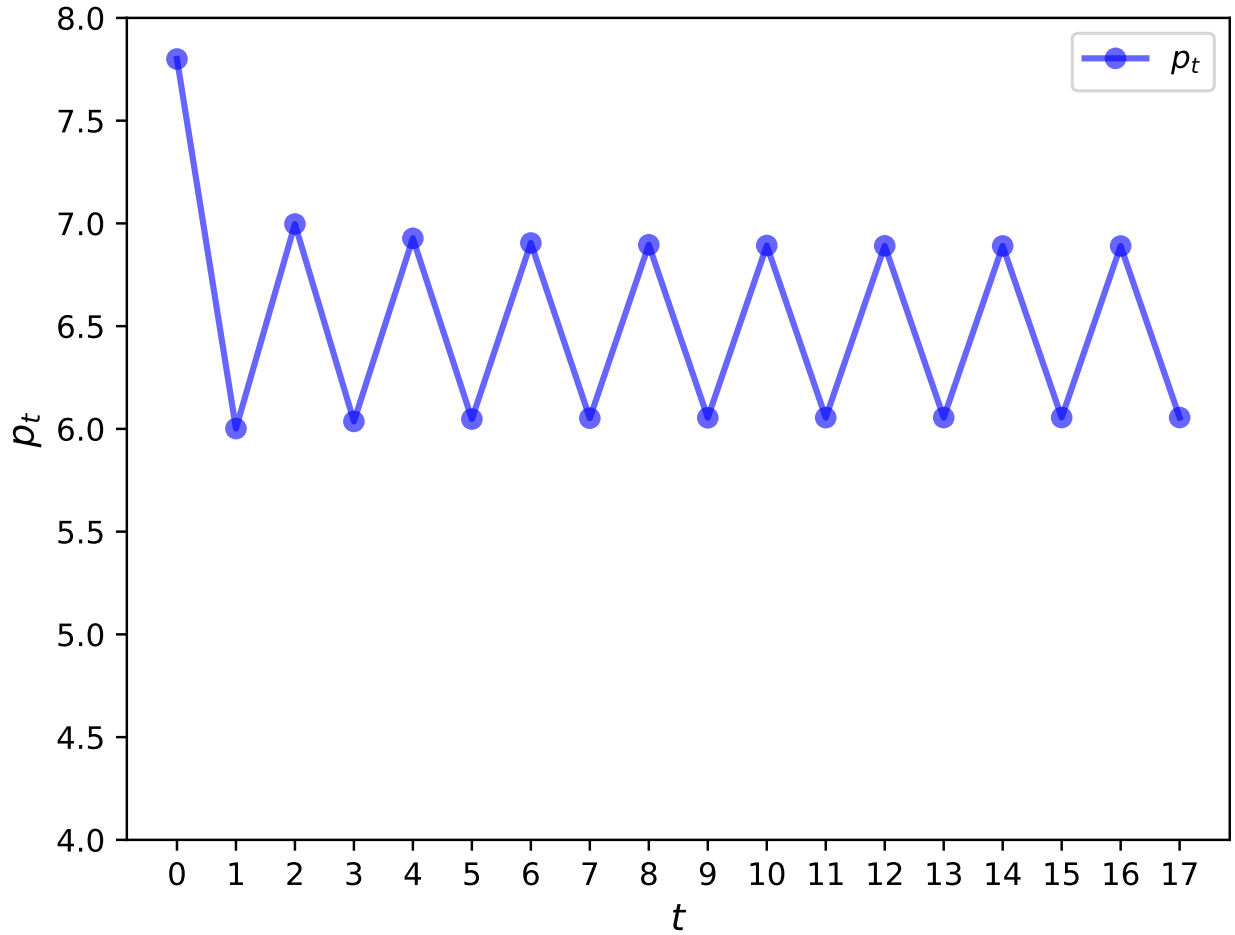
$$q^d(p_t) = q^s(p_{t-1})$$

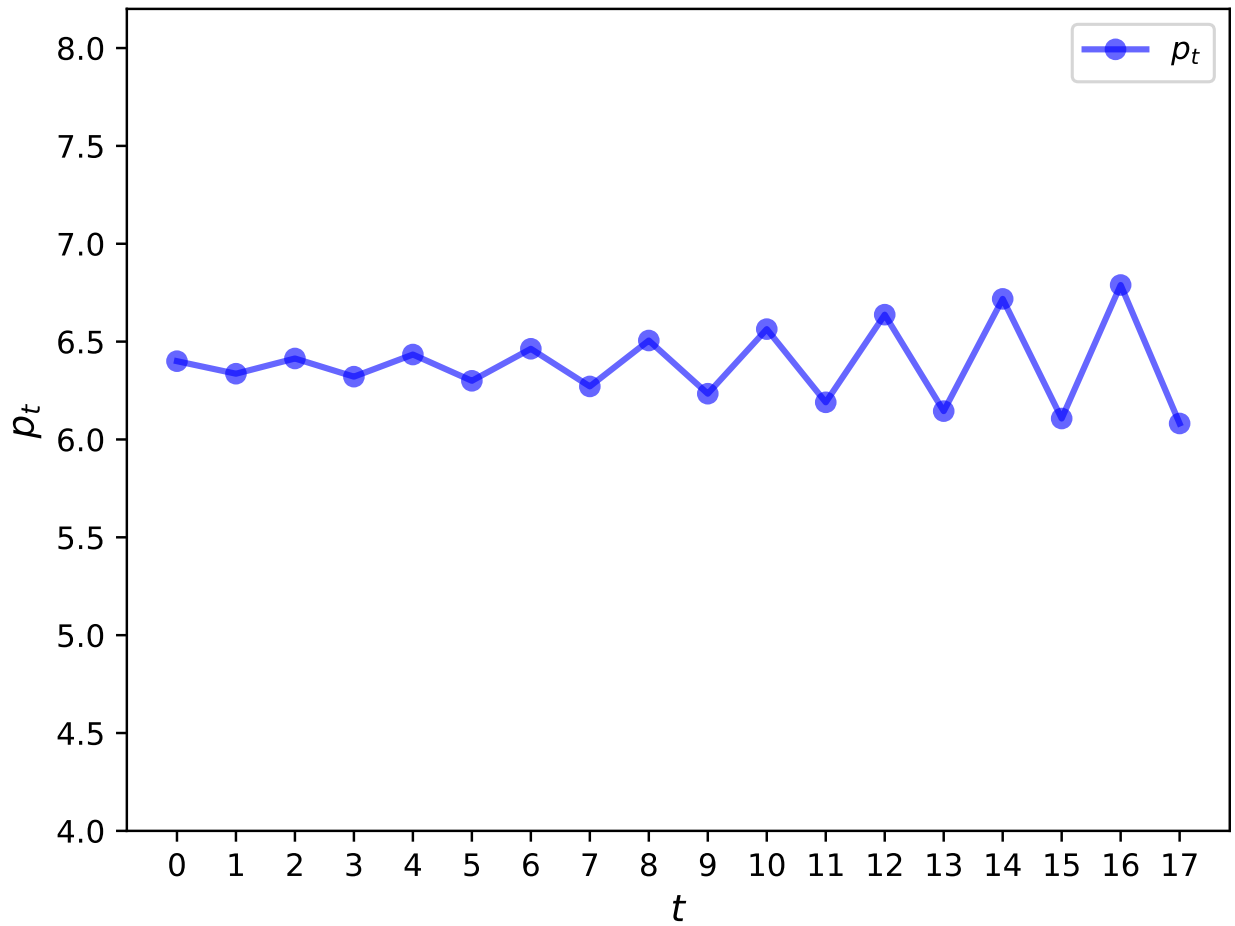
Solving for p_t gives

$$p_t = f(p_{t-1}) \quad \text{where} \quad f(p) = (q^d)^{-1}(q^s(p))$$

Now let's simulate...


8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
4.0
p_t
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
t
p_t


8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
4.0
p_t
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
t
p_t

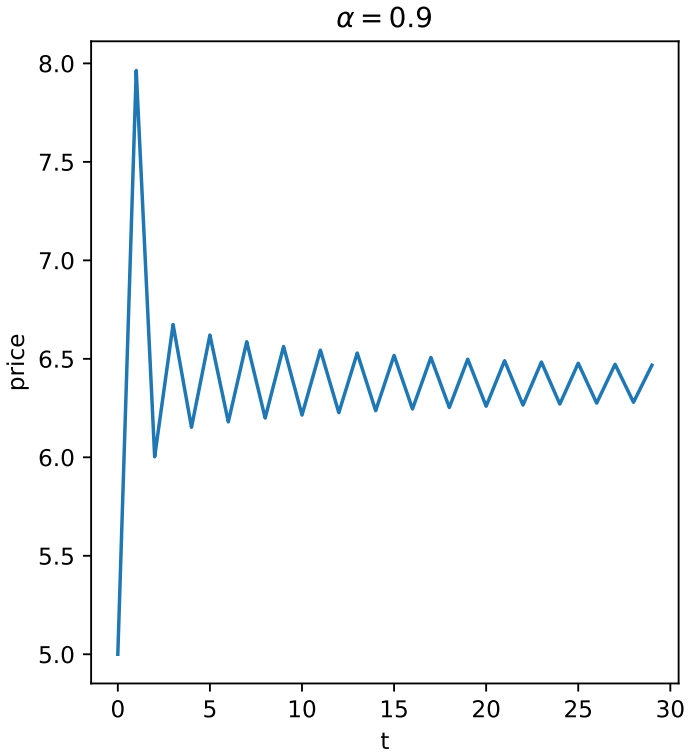

p_t
8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
4.0
p_t
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
t

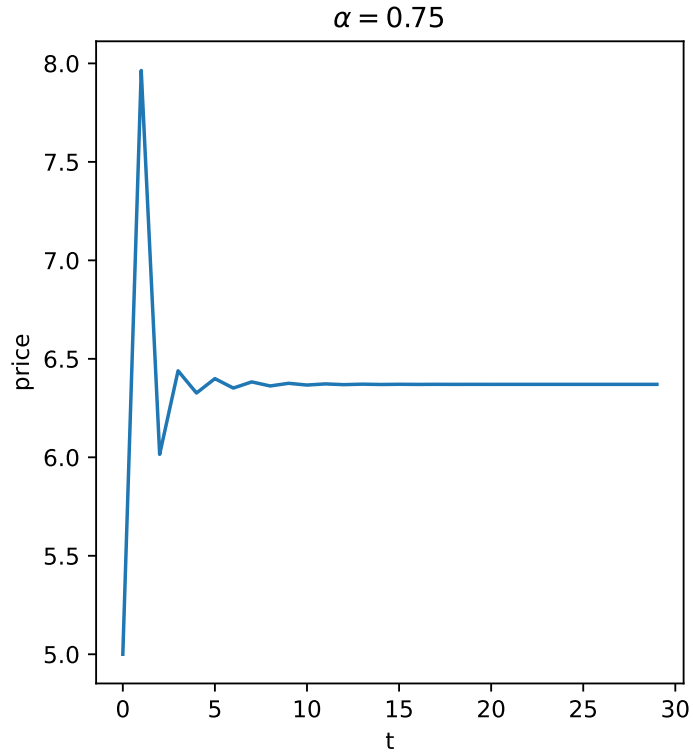
The model replicates cycles — but there are problems!

Predictions are **very sensitive** to how we model **expectations**

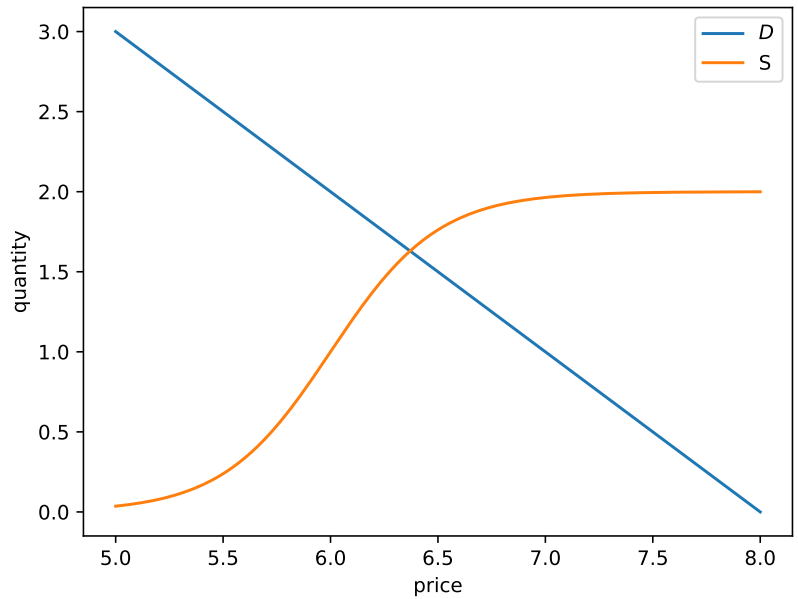
Example. Suppose we switch to $p^e_{t-1} = p^e_{t-2} + \alpha(p_{t-1} - p^e_{t-2})$

- Called "adaptive expectations"


$\alpha = 0.9$
price
t
8.0
7.5
7.0
6.5
6.0
5.5
5.0
0
5
10
15
20
25
30


$\alpha = 0.75$
price
t
8.0
7.5
7.0
6.5
6.0
5.5
5.0
0
5
10
15
20
25
30

Or we could use “rational expectations”



Lessons:

1. modeling human behavior is essential
2. the "right" way to model humans is unclear
3. predictions are very sensitive to their expectations
4. models are nonstationary because the way we predict is nonstationary

Summary: economic modeling is hard – but we shouldn't give up!

Scientific computing in economics

Anonymous consultant's report on HPC in economics:

Economists are relative newcomers to the field of computational sciences...

Economists have long been influenced by dogmatic tribalism...

It would appear that many (so called) 'theories' have been poorly (if at all!) proven...

Computational models in economics are still often simplistic...

Scientific computing in economics

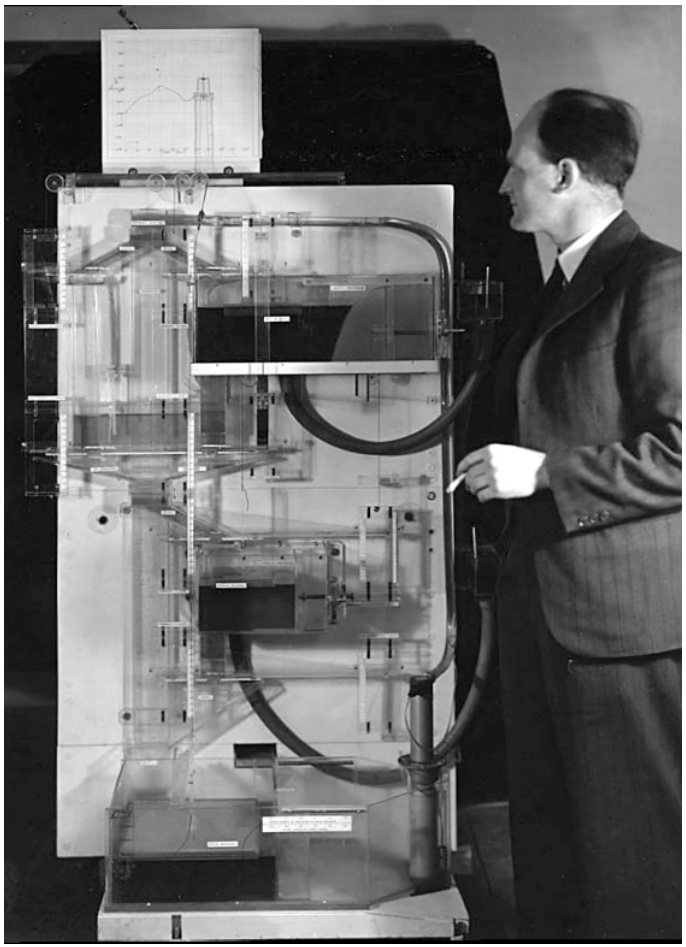
Anonymous consultant's report on HPC in economics:

Economists are relative newcomers to the field of computational sciences...

Economists have long been influenced by dogmatic tribalism...

It would appear that many (so called) 'theories' have been poorly (if at all!) proven...

Computational models in economics are still often simplistic...



Actually economists are pioneers
(William Phillips, 1949)

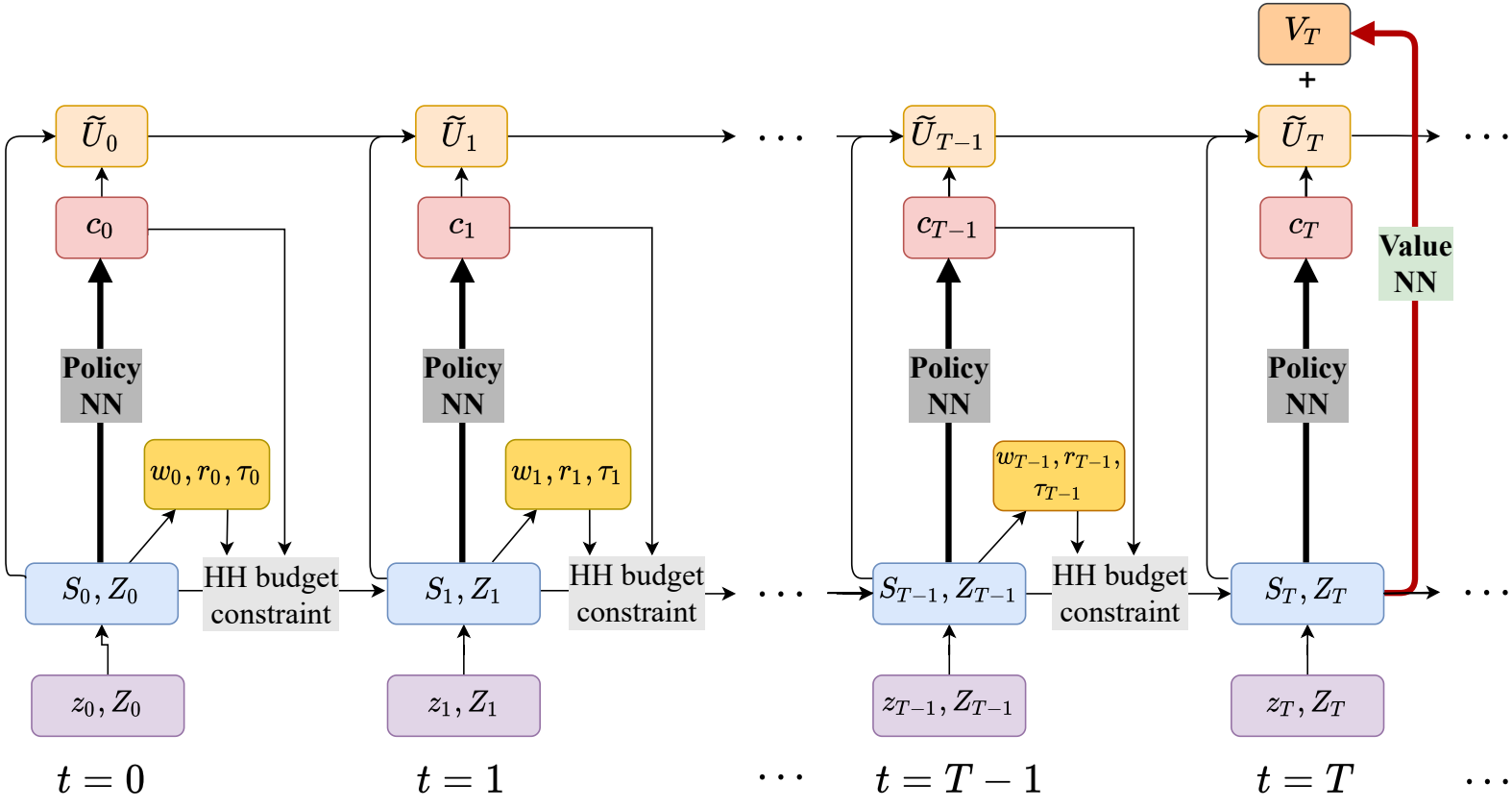
Modern scientific computing in economics

There are many examples of modern, sophisticated scientific computing in economics

Example. DeepHAM by Jiequn Han, Yucheng Yang, and Weinan E

- heterogeneous agent model
- firms and households linked by markets
- individual and aggregate risk
- general equilibium
- NNs to represent human reaction functions

KS DeepHAM


V_T
+
$\widetilde{U}_0$
$\widetilde{U}_1$
$\widetilde{U}_{T-1}$
$\widetilde{U}_T$
c_0
c_1
c_{T-1}
c_T
Policy NN
Policy NN
Policy NN
Policy NN
Value NN
w_0, r_0, τ_0
w_1, r_1, τ_1
$w_{T-1}, r_{T-1}, \tau_{T-1}$
S_0, Z_0
HH budget constraint
S_1, Z_1
HH budget constraint
S_{T-1}, Z_{T-1}
HH budget constraint
S_T, Z_T
z_0, Z_0
z_1, Z_1
z_{T-1}, Z_{T-1}
z_T, Z_T
$t = 0$
$t = 1$
$t = T - 1$
$t = T$

Trends in scientific computing

Technology is advancing rapidly along many dimensions

- improvements to algorithms
- new programming languages
- new hardware
- AI, etc.

What trends are affecting economics?

How are they changing economic research?

Trend 1: Proprietary → Open Source

Proprietary

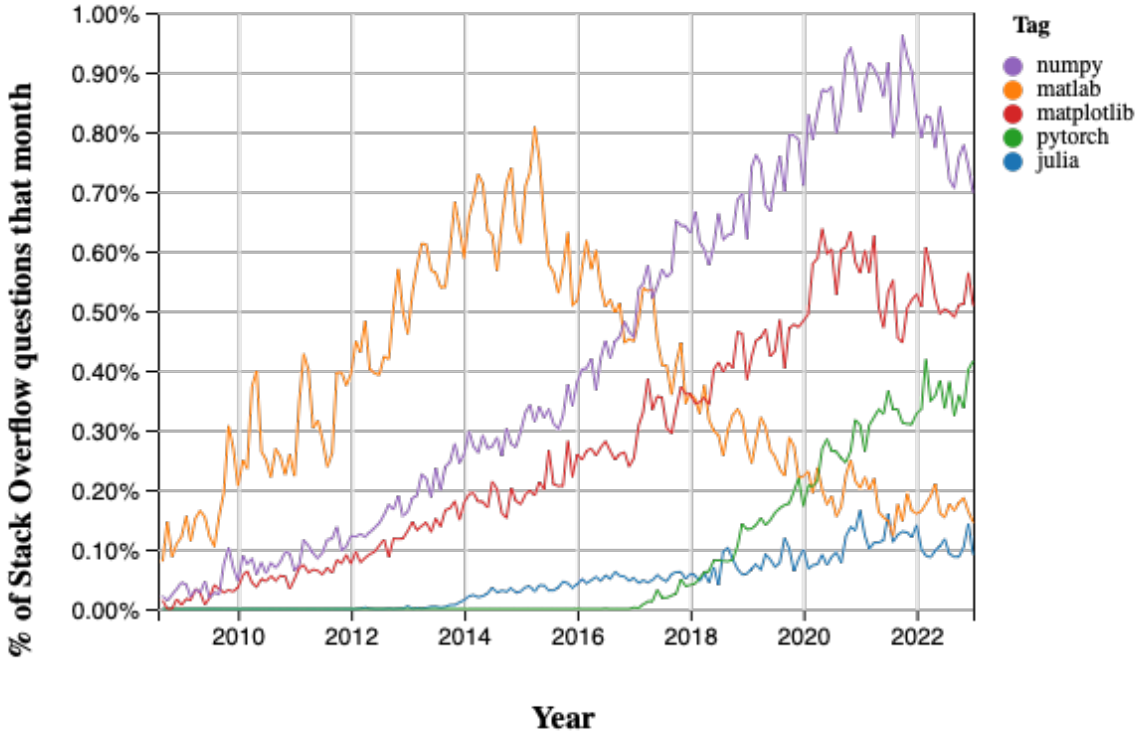
- Excel
- MATLAB, Mathematica
- STATA, Eviews, SPSS.

Open Source / Open Standard

- Python
- Julia
- R

closed and stable vs open and fast moving

Popularity:


% of Stack Overflow questions that month
1.00%
0.90%
0.80%
0.70%
0.60%
0.50%
0.40%
0.30%
0.20%
0.10%
0.00%
2010
2012
2014
2016
2018
2020
2022
Year
Tag
numpy
matlab
matplotlib
pytorch
julia

Trend 2: Low Level $\rightarrow$ High Level

Low level

- C/C++
- Fortran
- Assembly

High level

- Python
- Javascript
- PHP

Low level languages give us control

- control CPU
- control memory

High level languages give us

- abstraction
- automation
- flexibility, etc.

Example. $1 + 1$ in assembly

```
pushq   %rbp
movq    %rsp, %rbp
movl    $1, -12(%rbp)
movl    $1, -8(%rbp)
movl    -12(%rbp), %edx
movl    -8(%rbp), %eax
addl    %edx, %eax
movl    %eax, -4(%rbp)
movl    -4(%rbp), %eax
popq    %rbp
```

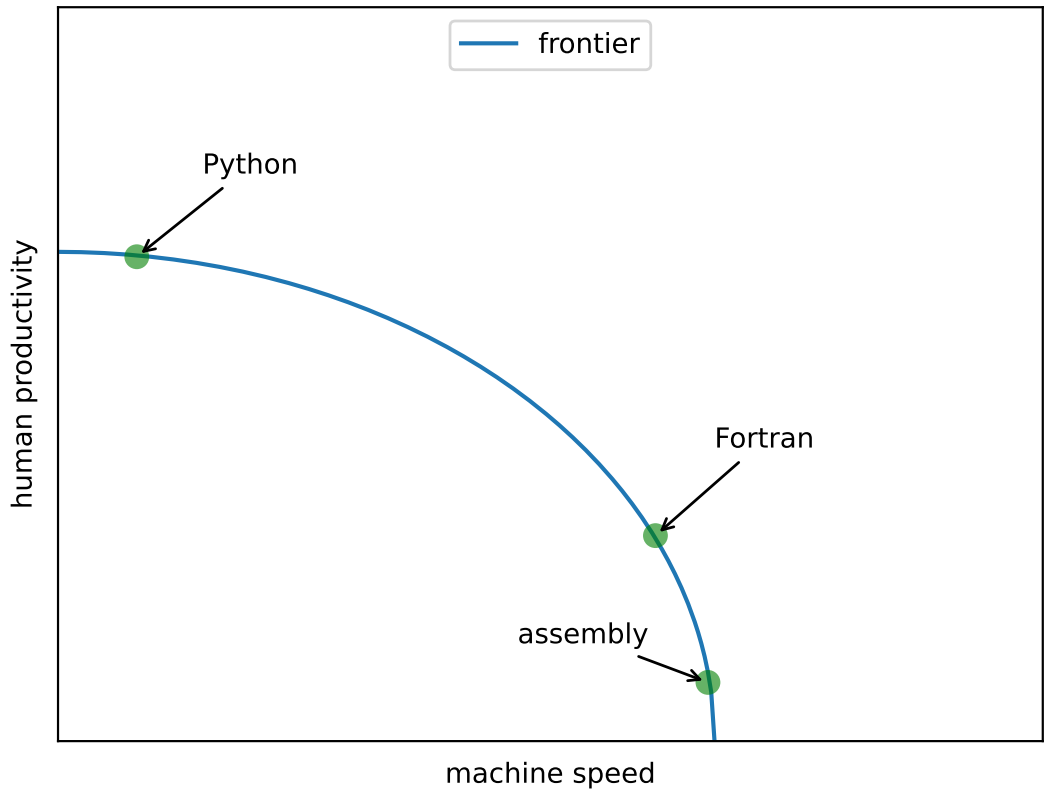
Example. $1 + 1$ in C

```c
#include <stdio.h>
int main() {
    int sum = 1 + 1;
    printf("1 + 1 = %d\n", sum);
    return 0;
}
```

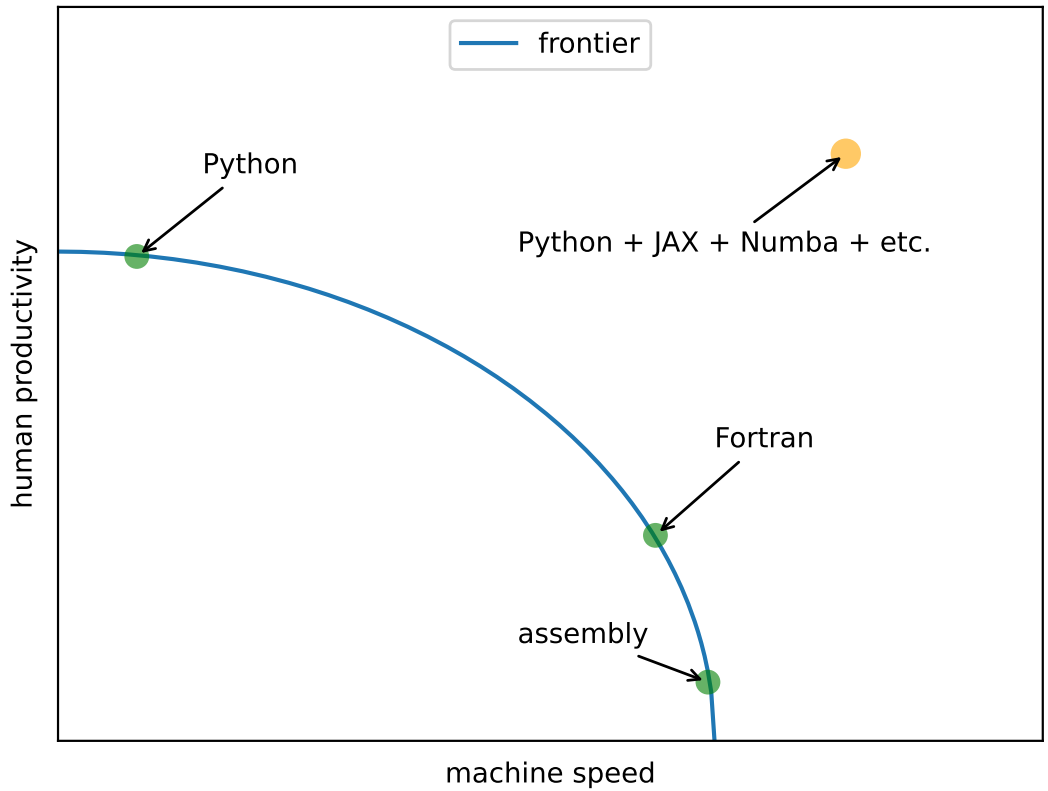
Example. $1 + 1$ in Python

```
sum = 1 + 1
print("1 + 1 = ", sum)
```

Trade-offs:


frontier
Python
Fortran
assembly
human productivity
machine speed

New trend — a shifting frontier!


frontier
Python
Python + JAX + Numba + etc.
Fortran
assembly
human productivity
machine speed

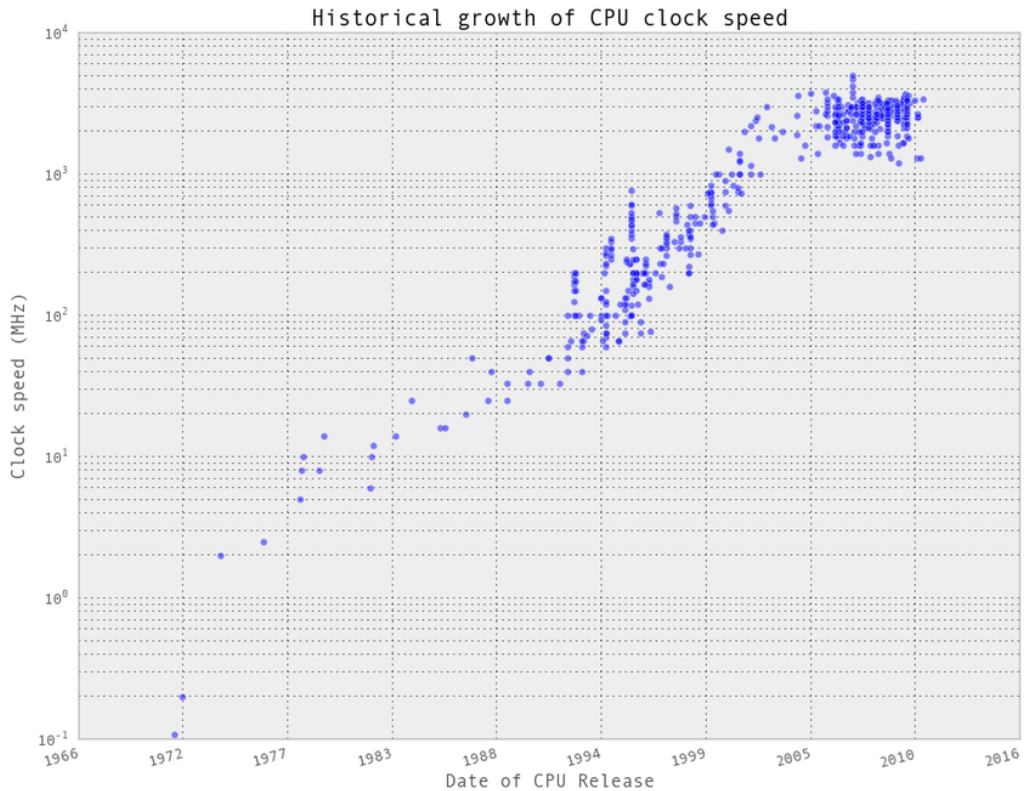
Example. `Numba` / `codon` generate fast machine code from Python

See code at

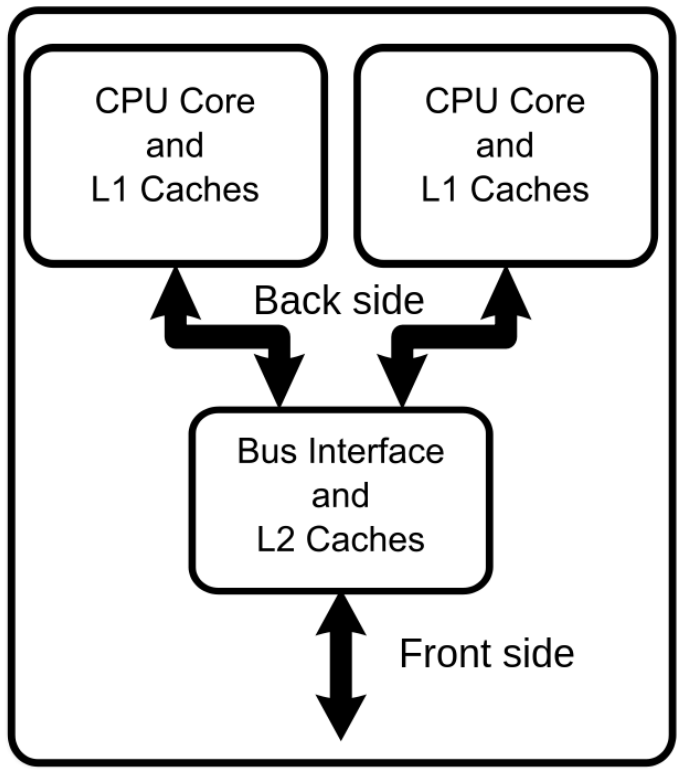
`https://github.com/QuantEcon/tokyo_college_2023/`

Trend 3: Parallelization

CPU frequency (clock speed) growth is slowing

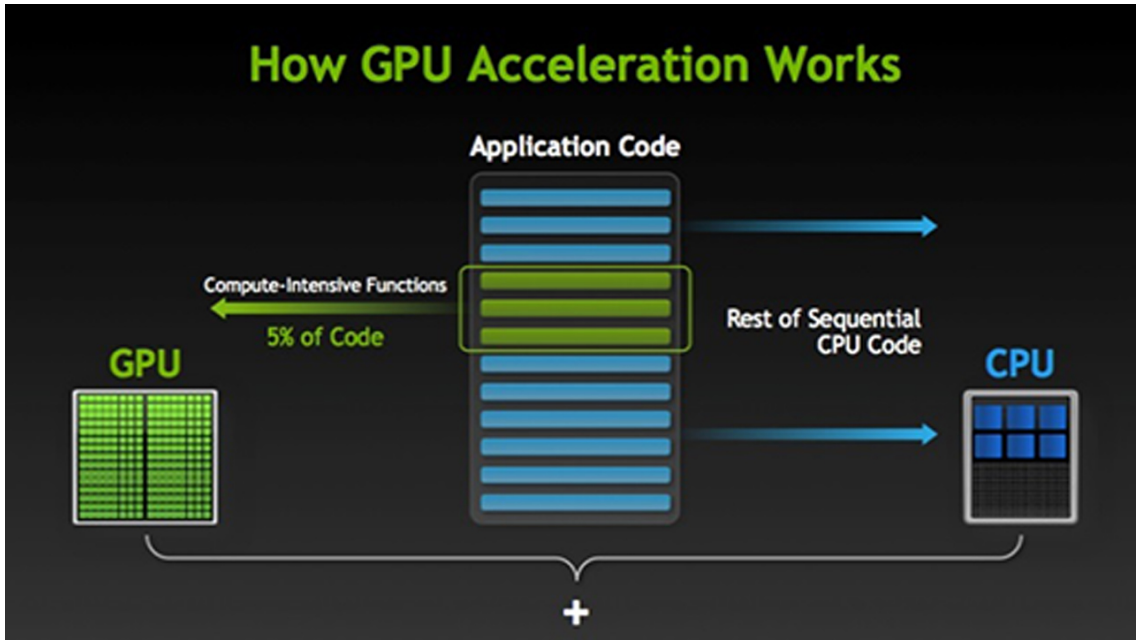


Chip makers have responded by developing multi-core processors



Source: Wikipedia

GPUs are becoming increasingly important



Applications: machine learning, deep learning, etc.

Example. JAX

See code at

https://github.com/QuantEcon/tokyo_college_2023/

The limits of computer power

Consider optimizing a function via brute force

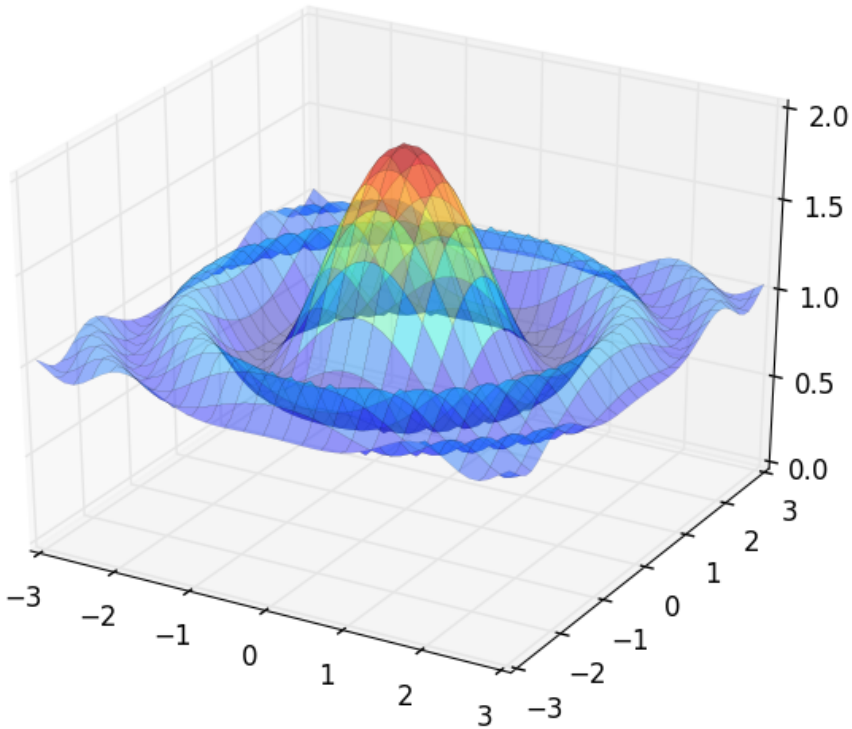


Figure: The function to maximize

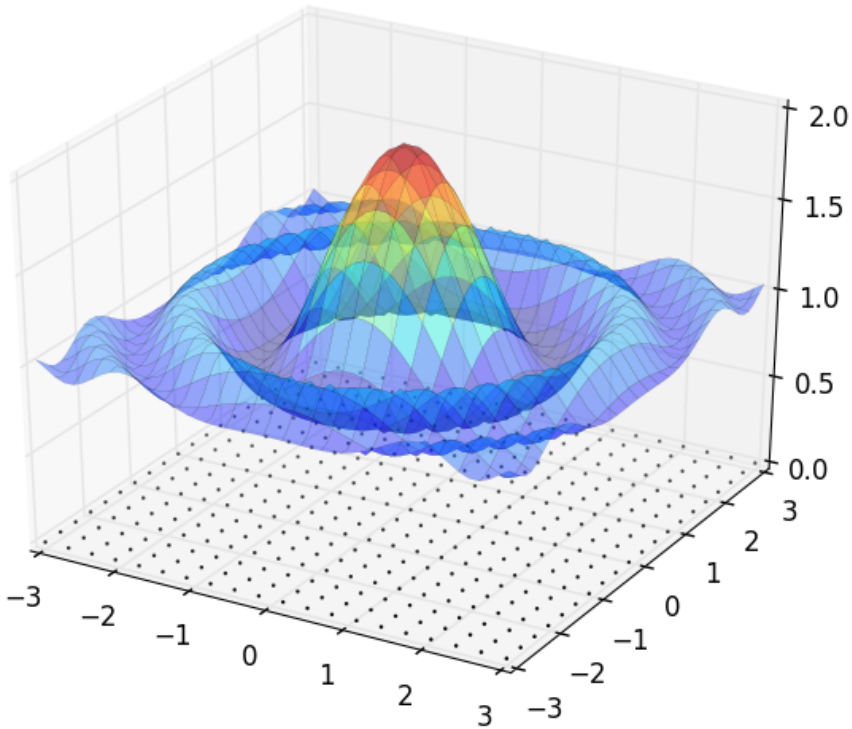


Figure: Grid of points to evaluate the function at

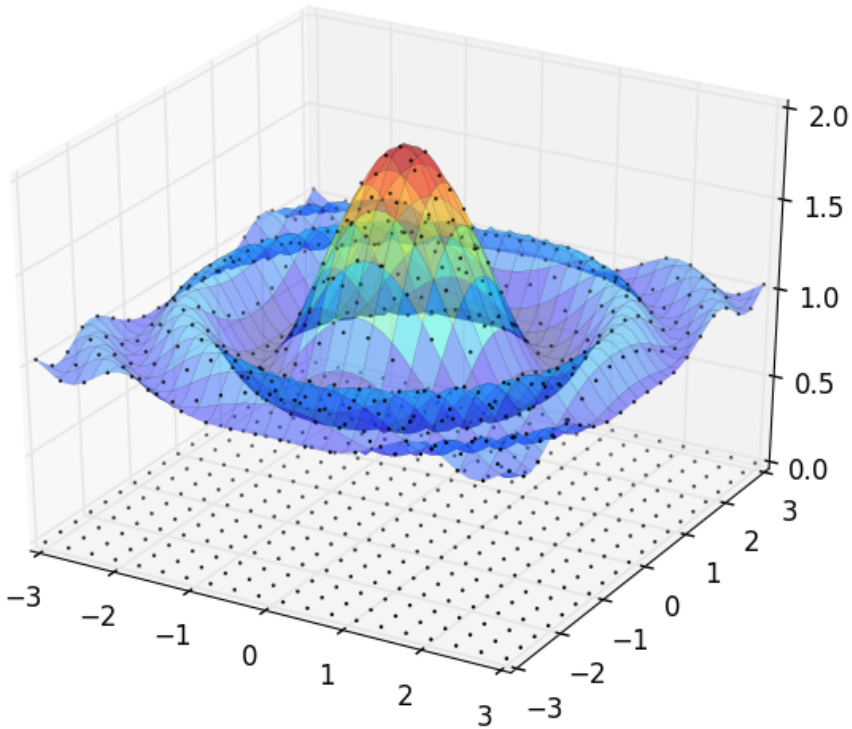


Figure: Evaluations

Grid size $= 20 \times 20 = 400$

Outcomes

- function evaluations $= 400$
- Time taken ≈ 0
- Max value $= 1.951$
- True maximum $= 2$

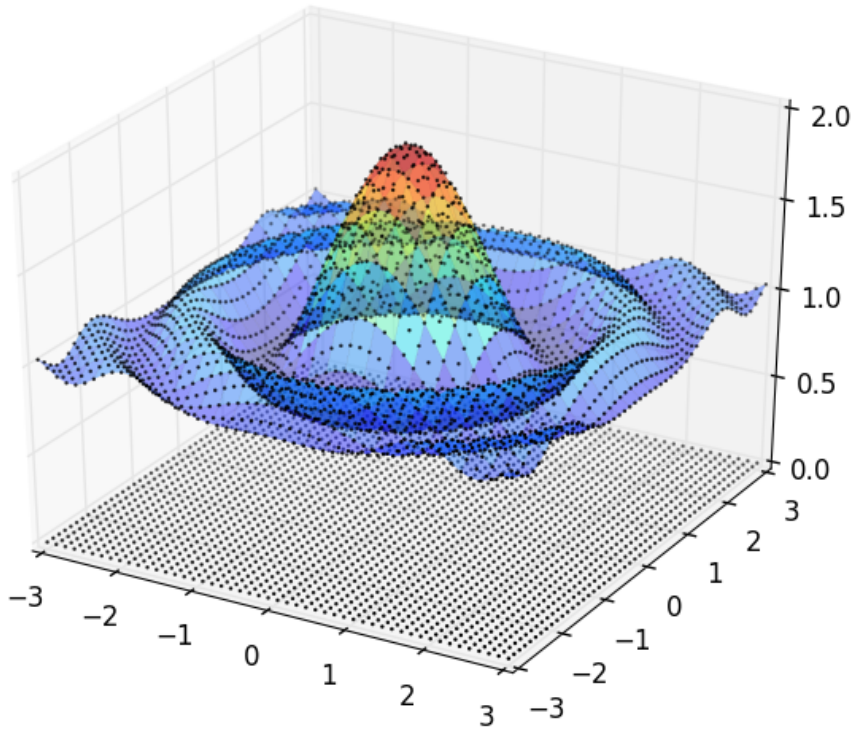


Figure: $50^2 = 2500$ evaluations

- function evaluations $= 50^2$
- Time taken $= 101$ microseconds
- Max value $= 1.992$
- True maximum $= 2$

But now suppose we hav more choice variables

- 3 vars: $\max_{x_1,x_2,x_3} f(x_1, x_2, x_3)$
- 4 vars: $\max_{x_1,x_2,x_3,x_4} f(x_1, x_2, x_3, x_4)$
- ...

If we have 50 grid points per variable and

- 2 variables then evaluations $= 50^2 = 2500$
- 3 variables then evaluations $= 50^3 = 125,000$
- 4 variables then evaluations $= 50^4 = 6,250,000$
- 5 variables then evaluations $= 50^5 = 312,500,000$
- ...

Example. Recent study: Optimal placement of drinks across vending machines in Tokyo

Approximate dimensions of problem:

- Number of choices for each variable $= 2$
- Number of choice variables $= 1000$

Hence number of possibilities $= 2^{1000}$

How big is that?

```
In [10]: 2**1000
Out[10]:
```
```
10715086071862673209484250490600018105614048117055336074437503883703510511249361224931983788156958581275946729175531468251871452856923140435984577574698574803934567774824230985421074605062371141877954182153046474983581941267398767559165543946077062914571196477686542167660429831652624386837205668069376
```

Suppose my machine evaluates about 10^9 choices per second

How long would that take?

```
In [16]: (2**1000 / 10**9) / 31556926  # In years
Out[16]:
3395478403651443492780079558636357072806789899958993494625396619335961465717339269652558613648540602869857073269915919013110292446394538059880920459330726574551199243812350729415493323101993883015713945697070264379864484033520491685142445099398167906015686216612651741700199135889415960
```

What about high performance computing?

- faster CPUs
- clusters of GPUs
- ...

Let's say speed up is 10^{12} (wildly optimistic)

```
In [19]: (2**1000 / 10**(9 + 12)) / 31556926
Out[19]:
339547840365144349278007955863635707280678989995899349462539661933596146571733926965255861364854060286985707326991591901311029244639453805988092045933072657455119924381235072941549332310199388301571394569707026437986448403352049168514244509939816790601568621661265174170019
```

For comparison:

```
In [20]: 5 * 10**9 # Expected lifespan of sun
Out[20]: 5000000000
```

Message: There are serious limits to computation

What's required is clever analysis

Exploit all available structure

Good algorithms are still crucial

Algorithms interact with advances in hardware/software

What about machine learning & AI?

Current gen AI cannot solve most economic problems

Consider the current global inflationary episode

- Many aspects are unprecedented

What will be the impact of policy interventions?

- Often we have no data

An ongoing need for careful mathematical modeling