

# Python and the AI Revolution

Chase Coleman and John Stachurski

March 2024

# Topics

We will discuss

- Deep learning and AI
- AI-driven scientific computing
- Where are we heading?
- How will that impact economic modeling for policy work?

# AI-driven scientific computing

AI is changing the world

- LLMs
- image processing and computer vision
- speech recognition, translation
- scientific knowledge discovery
- forecasting and prediction

Plus killer drones, skynet, etc....

**Key point:** vast investment in AI is changing the choice set for **all** scientific coders

# AI-driven scientific computing

AI is changing the world

- LLMs
- image processing and computer vision
- speech recognition, translation
- scientific knowledge discovery
- forecasting and prediction

Plus killer drones, skynet, etc....

**Key point:** vast investment in AI is changing the choice set for **all** scientific coders

# AI-driven scientific computing

AI is changing the world

- LLMs
- image processing and computer vision
- speech recognition, translation
- scientific knowledge discovery
- forecasting and prediction

Plus killer drones, skynet, etc....

**Key point:** vast investment in AI is changing the choice set for **all** scientific coders

## Key players

- OpenAI / Microsoft
- Google (Google Research, Google DeepMind)
- Meta
- Anthropic, etc.

## Platforms / libraries

- PyTorch (ChatGPT, Meta's LLaMA 2, Stable Diffusion)
- Google JAX (Google's Gemini)
- Tensorflow, Keras, Mojo?

# Deep learning in two slides

Supervised deep learning: find a good approximation to an unknown functional relationship

$$y = f(x)$$

## Examples.

- $x$  = sequence of words,  $y$  = next word
- $x$  = weather sensor data,  $y$  = max temp tomorrow

Problem:

- observe  $(x_i, y_i)_{i=1}^n$  and seek  $f$  such that  $y_{n+1} \approx f(x_{n+1})$

Nonlinear regression: minimize the empirical loss

$$\ell(\theta) := \sum_{i=1}^n (y_i - f_{\theta}(x_i))^2 \quad \text{s.t.} \quad \theta \in \Theta$$

But what is  $\{f_{\theta}\}_{\theta \in \Theta}$ ?

In the case of ANNs, we consider all  $f_{\theta}$  having the form

$$f_{\theta} = \sigma \circ A_1 \circ \cdots \circ \sigma \circ A_{k-1} \circ \sigma \circ A_k$$

where

- $A_i x = W_i x + b_i$  is an affine map
- $\sigma$  is a nonlinear “activation” function



Nonlinear regression: minimize the empirical loss

$$\ell(\theta) := \sum_{i=1}^n (y_i - f_{\theta}(x_i))^2 \quad \text{s.t.} \quad \theta \in \Theta$$

But what is  $\{f_{\theta}\}_{\theta \in \Theta}$ ?

In the case of ANNs, we consider all  $f_{\theta}$  having the form

$$f_{\theta} = \sigma \circ A_1 \circ \cdots \circ \sigma \circ A_{k-1} \circ \sigma \circ A_k$$

where

- $A_i x = W_i x + b_i$  is an affine map
- $\sigma$  is a nonlinear “activation” function

Nonlinear regression: minimize the empirical loss

$$\ell(\theta) := \sum_{i=1}^n (y_i - f_{\theta}(x_i))^2 \quad \text{s.t.} \quad \theta \in \Theta$$

But what is  $\{f_{\theta}\}_{\theta \in \Theta}$ ?

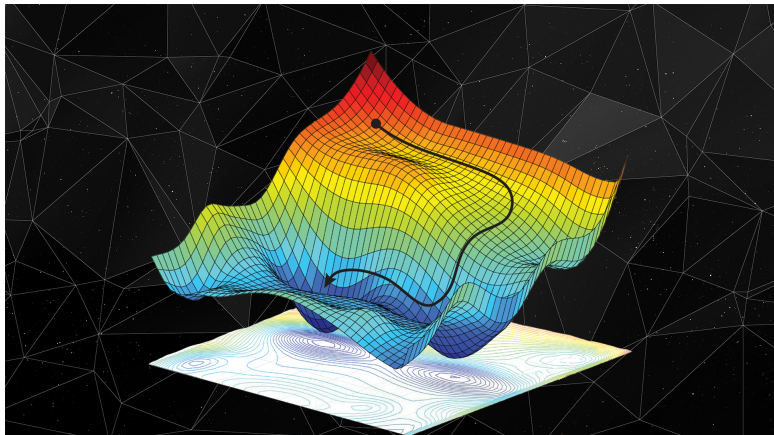
In the case of ANNs, we consider all  $f_{\theta}$  having the form

$$f_{\theta} = \sigma \circ A_1 \circ \cdots \circ \sigma \circ A_{k-1} \circ \sigma \circ A_k$$

where

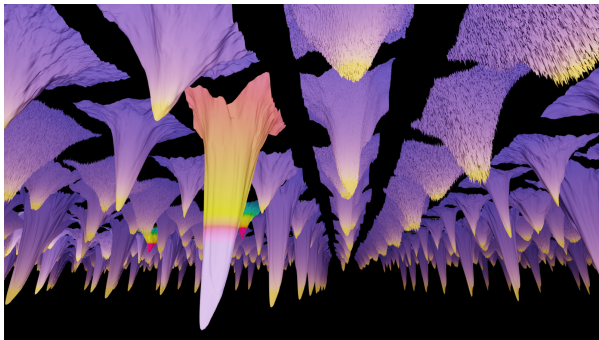
- $A_i x = W_i x + b_i$  is an affine map
- $\sigma$  is a nonlinear “activation” function

Minimizing a smooth loss functions – what algorithm?



Source: <https://danielkhv.com/>

Deep learning:  $\theta \in \mathbb{R}^d$  where  $d = ?$



Source: <https://losslandscape.com/gallery/>

But what about the curse of dimensionality!???

# Software



## Core elements

- automatic differentiation (for gradient descent)
- parallelization (GPUs! — how many?)
- Compilers / JIT-compilers

Crucially, these components are all integrated

- autodiff is JIT compiled
- JIT compiled functions are automatically parallelized
- etc.

## Core elements

- automatic differentiation (for gradient descent)
- parallelization (GPUs! — how many?)
- Compilers / JIT-compilers

Crucially, these components are all integrated

- autodiff is JIT compiled
- JIT compiled functions are automatically parallelized
- etc.





---

```
import jax.numpy as jnp
from jax import grad, jit
```

```
def f(params, x):
    for W, b in params:
        y = jnp.dot(W, x) + b
        x = jnp.tanh(y)
    return y
```

```
def loss(params, x, y):
    return jnp.sum((f(params, x) - y)**2)
```

```
grad_loss = jit(grad(loss))  # Now use gradient descent
```

---

Source: Google JAX readthedocs

# Hardware



“NVIDIA today announced its next-generation AI supercomputer — the NVIDIA DGX SuperPOD powered by GB200 Grace Blackwell Superchips — for processing trillion-parameter models for superscale generative AI training and inference workloads.

Featuring a new, highly efficient, liquid-cooled rack-scale architecture, the DGX SuperPOD provides 11.5 exaflops of AI supercomputing and 240 terabytes of fast memory.”

“NVIDIA supercomputers are the factories of the AI industrial revolution.” – Jensen Huang

## Example: Weather forecasting

“ECMWF’s weather forecasting model is considered the gold standard for medium-term weather forecasting...”

Google DeepMind claims to now beat it 90% of the time...

“Traditional forecasting models are big, complex computer algorithms based on atmospheric physics and take hours to run. AI models can create forecasts in just seconds.”

Source: MIT Technology Review

## Example: Weather forecasting

“ECMWF’s weather forecasting model is considered the gold standard for medium-term weather forecasting...”

Google DeepMind claims to now beat it 90% of the time...

“Traditional forecasting models are big, complex computer algorithms based on atmospheric physics and take hours to run. AI models can create forecasts in just seconds.”

Source: MIT Technology Review

## Relevant to economics?

Deep learning provides massively powerful pattern recognition

**But** macroeconomic data is

- extremely limited
- generally nonstationary
- sensitive to policy changes (Lucas critique)

Relevant to economics?

Deep learning provides massively powerful pattern recognition

**But** macroeconomic data is

- extremely limited
- generally nonstationary
- sensitive to policy changes (Lucas critique)



Relevant to economics?

Deep learning provides massively powerful pattern recognition

**But** macroeconomic data is

- extremely limited
- generally nonstationary
- sensitive to policy changes (Lucas critique)

## My view

- Policy-centric macroeconomic modeling will survive much longer than traditional weather forecasting
- Deep learning is yet to prove itself as a “better” approach to numerical methods

And yet,

- the AI computing revolution is generating tools that are enormously beneficial for macroeconomic modeling
  - autodiff, JIT compilers, parallelization, GPUs, etc.
- We can take full advantage of them right now

And that's exactly what we're going to do!

## My view

- Policy-centric macroeconomic modeling will survive much longer than traditional weather forecasting
- Deep learning is yet to prove itself as a “better” approach to numerical methods

And yet,

- the AI computing revolution is generating tools that are enormously beneficial for macroeconomic modeling
  - autodiff, JIT compilers, parallelization, GPUs, etc.
- We can take full advantage of them right now

And that's exactly what we're going to do!

## My view

- Policy-centric macroeconomic modeling will survive much longer than traditional weather forecasting
- Deep learning is yet to prove itself as a “better” approach to numerical methods

And yet,

- the AI computing revolution is generating tools that are enormously beneficial for macroeconomic modeling
  - autodiff, JIT compilers, parallelization, GPUs, etc.
- We can take full advantage of them right now

And that's exactly what we're going to do!