AI for Economics Research

From Theory to Practice

Max Ghenis · PolicyEngine

University of South Carolina · Economics PhD Class

About Me

Max Ghenis

- **CEO & Founder**, PolicyEngine
- **Mission**: Make public policy analysis accessible to everyone
- **Background**: Economics, Data Science, Policy Research
- **Passion**: Using AI to democratize policy analysis

PolicyEngine

What we do:

- Free, open-source tax-benefit microsimulation
- Model reforms instantly
- Household & societal impacts
- US, UK, and Canada

AI Integration:

- Enhanced CPS data with LLMs
- Household AI assistant
- GPT-powered analysis

The Journey: From ChatGPT to Claude Code

📅 One Year Ago

- Saw paper on LLMs simulating survey responses (r=0.9)
- Started exploring: *"Can LLMs simulate economic behavior?"*

🎯 The ETI Paper

- *"What Can LLMs Teach Us About the ETI?"*
- With Jason Debacker (your professor!)
- Presented at National Tax Association

✈️ The Vegas Airport Moment

- 6 months ago: Claude Code launches
- Building on a layover
- First merge conflict: `1897111601662321090`

Test-Driven Development: The Secret Weapon

Why TDD + Claude Code = 🚀

```python
def test_eti_calculation():
    """Test elasticity of taxable income calculation"""
    pre_tax_income = 100000
    post_tax_income = 95000
    tax_rate_1 = 0.3
    tax_rate_2 = 0.35

    eti = calculate_eti(pre_tax_income, post_tax_income,
                        tax_rate_1, tax_rate_2)

    assert abs(eti - (-0.69)) < 0.01  # Expected ETI
```

Red-Green-Refactor Cycle

1. **Write the test** (it fails - RED)
2. **Claude Code writes implementation** (test passes - GREEN)
3. **Refactor together** (improve code quality)

TDD in Action: Grant Proposals

```python
def test_grant_proposal_word_count():
    """Ensure grant sections meet word limits"""
    proposal = load_grant_proposal()

    # Test executive summary
    exec_summary = proposal['executive_summary']
    word_count = len(exec_summary.split())
    assert word_count ⩽ 250, f"Executive summary: {word_count} words (max 250)"

    # Test project description
    description = proposal['project_description']
    word_count = len(description.split())
    assert word_count ⩽ 2000, f"Description: {word_count} words (max 2000)"
```

Real example: Used TDD to ensure our grant proposals met all requirements!

Evolution of My Workflow

Before Claude Code

- ChatGPT for help
- Manual copy-paste
- Context switching
- ~2-3x productivity boost

With Claude Code + TDD

- Integrated terminal workflow
- Automatic file management
- **5-10x productivity**
- $200/mo Claude Max = best investment

The Game Changer: `--dangerously-skip-permissions`

- 30+ minute sessions
- Complete project transformations
- *"AGI is already here"*

Live Demo: Economic Analysis with Claude Code

Let’s analyze Fed minutes in real-time!

```
# We'll build this together with Claude Code
import pandas as pd
import requests
from transformers import pipeline

def analyze_fed_sentiment():
    """Analyze Federal Reserve meeting minutes sentiment"""
    # Fetch latest Fed minutes
    # Apply sentiment analysis
    # Calculate hawkish/dovish score
    # Visualize results
    pass
```

Using TDD approach:

1. First, write tests for what we want
2. Let Claude Code implement
3. Verify with real Fed data

The PolicyEngine Journey with AI

Projects Built with Claude Code

Enhanced CPS Launch

- LLM-powered data imputation
- Improved microsimulation accuracy
- policyengine.org/us/research/enhanced-cps-launch

Household AI Assistant

- Natural language policy queries
- Personalized impact analysis
- policyengine.org/us/research/us-household-ai

GPT Analysis Integration

- Automated policy explanations
- Dynamic report generation
- policyengine.org/us/research/gpt-analysis

LLM-ETI Research

- Simulating economic responses
- Replicating elasticity studies
- github.com/MaxGhenis/llm-eti

My Claude Code Evolution: A Tweet Journey

First Steps

Screenshots: Building This Presentation

github-issue.png


GitHub Issue Creation

Claude Code creating issues automatically

workflow-screenshot.png


Workflow

The full development workflow

claude-code-building.png


Building with Claude Code

Using Claude Code to build this very presentation!

Real Economics Research: The ETI Paper

"What Can LLMs Teach Us About the ETI?"

Key Findings:

- LLMs can simulate survey responses with **r=0.9** accuracy
- Successfully replicated **elasticity** studies
- Cost: ~$50 in API calls vs $500K+ for traditional **survey experiments**

Methodology with TDD:

```
def test_llm_survey_response():
    """Test LLM's ability to simulate survey responses"""
    prompt = create_survey_prompt(income=75000, tax_rate=0.25)
    response = llm.generate(prompt)

    # Validate response is within expected bounds
    assert 0 ⩽ response.labor_supply_change ⩽ 1.0
    assert response.confidence ⩾ 0.8
```

Turn Everything Into Software

The Philosophy

> "TURN EVERYTHING INTO SOFTWARE. I genuinely think AGI is already here, and it's Claude Code Opus."

What This Means:

1. **Grant proposals** → Interactive websites
2. **Research papers** → Reproducible notebooks
3. **Data analysis** → Automated pipelines
4. **Literature reviews** → Living documents

Real Examples:

- **Office evaluation** → Interactive comparison site
  - Pricing, travel times, amenities
  - Built faster than writing a doc!
- **Grant proposals** → Interactive websites

API Integrations: Your Research Assistant

Connect Everything

Data Sources

- **FRED API**: Economic indicators
- **World Bank**: Development data
- **Census**: Demographic data
- **GitHub API**: Version control

Productivity Tools

- **Google Drive**: Document management
- **Calendar**: Schedule optimization
- **Email**: Automated responses
- **Social Media**: Research dissemination

Example: Automated Data Pipeline

```
@test
def test_fred_data_fetch():
    data = fetch_fred_series('UNRATE')
    assert len(data) > 0
    assert 'date' in data.columns
```

For Academic Economists: Your Roadmap

Map the Research Process

1. **Literature Review** → Automated paper summaries
2. **Data Collection** → API integrations with TDD
3. **Cleaning & Analysis** → Test-driven transformations
4. **Econometrics** → Validated model specifications
5. **Visualization** → Interactive, reproducible charts
6. **Writing** → LaTeX + Markdown + Web
7. **Peer Review** → Simulated referee reports

Why Not Also:

- Build a web version with **Jupyter Book**
- Create interactive visualizations
- Simulate referee feedback
- Auto-improve based on reviews

Live Coding: Build an Economic Analysis

Let’s create a labor supply analysis with TDD!

```
# Start with the test
def test_labor_supply_elasticity():
    """Test labor supply response to tax changes"""
    # Given
    baseline_hours = 40
    wage = 25
    tax_increase = 0.05

    # When
    new_hours = calculate_labor_supply(
        baseline_hours, wage, tax_increase
    )

    # Then
    elasticity = (new_hours - baseline_hours) / baseline_hours
    assert -0.3 ⩽ elasticity ⩽ -0.1  # Expected range
```

Let Claude Code implement the economics!

The Ultimate Challenge: 100% Claude Code

My New Rule: Never Write Code by Hand

The Challenge:

- **NO manual coding** - not even variable names
- **NO docs** - build interactive sites instead
- **NO emails** - let Claude Code draft them
- **NO calendar management** - API integration
- **NO ChatGPT/Claude.ai** - only Claude Code

Example: Office Space Evaluation

Instead of a Google Doc → Built dc-office-comparison

- Pricing comparisons
- Commute times for each team member
- Tech/econ focus ratings
- Amenity scores **Time saved: 50%+**

The One Week Challenge

You have ONE WEEK before throttling begins!

Your Mission:

1. **Install Claude Code** with Opus 4
2. **Start with TDD** - write tests first
3. **Build something ambitious** - a full research project
4. **Use --dangerously-skip-permissions** for long sessions
5. **Transform your workflow** permanently

Ideas to Try:

- Replicate a famous paper
- Build an interactive textbook
- Create a policy simulator
- Automate your research pipeline

> "Make it your life's most productive week!"

Resources & Next Steps

Get Started Today

Tools:

- **Claude Code**: claude.ai/code
- **PolicyEngine**: policyengine.org
- **LLM-ETI Paper**: github.com/MaxGhenis/llm-eti

Connect:

- **Twitter/X**: @MaxGhenis
- **GitHub**: github.com/MaxGhenis
- **PolicyEngine**: @PolicyEngine

Remember:

- **Test-Driven Development** is your superpower
- **Turn everything into software**
- **The best $200/mo you'll ever spend**

Let’s explore the future of economics research together!

Max Ghenis

max@policyengine.org