# Tapping the Supercomputer Under Your Desk: Solving Dynamic Equilibrium Models with Graphics Processors

## (Companion Webpage)

Eric M. Aldrich (Duke University)
Jesús Fernández-Villaverde (University of Pennsylvania)
A. Ronald Gallant (Duke University and New York University)
Juan F. Rubio-Ramírez (Duke University and Federal Reserve Bank of Atlanta)

### Abstract (Full Paper) (Slides)

This paper shows how to build algorithms that use graphics processing units (GPUs) installed in most modern computers to solve dynamic equilibrium models in economics. In particular, we rely on the compute unified device architecture (CUDA) of NVIDIA GPUs. We illustrate the power of the approach by solving a simple real business cycle model with value function iteration. We document improvements in speed of around 200 times and suggest that even further gains are likely.

### Discussion

This paper was recently mentioned on the NEP-DGE blog. We invite you to contribute to the discussion.

### Code

Complete archive of the code necessary to reproduce the results in the paper (last update: 21 April 2010). This software was developed on a Linux machine using the 64-bit CUDA Toolkit 2.2 for Redhat Enterprise Linux 5.3. The code is not guaranteed to work with later versions of the CUDA toolkit or with other platforms.

### CUDA Installation (July 2010)

To download the latest CUDA Toolkit, visit the CUDA Toolkit Release Archive. The CUDA developer drivers, CUDA toolkit, GPU Computing SDKs and all requisite documentation can be downloaded from this webpage. The drivers, toolkit and documentation are available in 32-bit and 64-bit versions for Windows, Linux and Mac OS X.

As of July 2010, the latest release of the CUDA toolkit is version 3.1, which can be found at the CUDA Toolkit 3.1 webpage. To begin the installation process, download the "Getting Started Guide" which is available for the Windows, Linux and OS X (for example: Getting Started Guide for Mac). This document provides step-by-step instructions for verifying a CUDA capable machine, installing the developer drivers and toolkit and running the SDKs.

A MATLAB plug-in for CUDA is available and may be of interest to researchers who predominantly use MATLAB. The authors of the paper cited above have not used this plug-in and did not use MATLAB for the results in their paper (and hence cannot provide any advice on how to use the plug-in).

As mentioned above, the code for the paper was developed on a Linux machine using the 64-bit CUDA Toolkit 2.2 for Redhat Enterprise Linux 5.3 and is not guaranteed to work with later versions of the CUDA toolkit or with other platforms. To run the code and replicate the results of the paper,

- Install version 2.2 of the CUDA toolkit with corresponding developer drivers on a Linux machine.
- Download the TAR archive of the code to the test machine.
- Open a shell and navigate to the directory where the TAR archive is located. At the prompt, type "tar -xzvf vfi-2010-04-21.tar".
- At the prompt, type the following commands in sequence (with a carriage return between commands)
  - cd VFI
  - make
  - vfi

  The final command will run the code and print timing results to the display. Changes to various parameters of interest can be made in the file "globalvars.cpp".