



Multi-Institutional Collaborations for Materials Research and Learning

AAAS Annual Meeting: Science Without Borders

*Teaching and Learning in the Digital Age: Reliable
Resources across the Disciplines*

Washington DC, Feb. 20, 2011

**Laura Bartolo
Kent State University**



Outline

- Introduction to MatDL/MatForge
- MatForge
 - Research Codes
 - Research-based teaching resources
 - Script service
- Online Demo
- Concluding remarks

MatDL/MatForge



Materials Pathway



Supported by NSF & NSDL

Repository **MatForge** MatDL Wiki Virtual Labs Other Services Participate



MatForge
Use open source materials-related codes in your research and teaching.
[Go to MatForge](#) →

Overview
MatDL Pathway provides content and services for materials students, faculty, and researchers. [More](#) →

Professional Societies


Highlight
MatForge

FiPy: A Finite Volume PDE Solver (Python)
[Go to Resource](#) →

News

- MatDL Presents at 1st World Congress on ICME
- MatDL/MatForge Demo at 2011 TMS Annual Meeting
- [More](#) →

KENT STATE UNIVERSITY NIST National Institute of Standards and Technology MIT Massachusetts Institute of Technology UNIVERSITY OF MICHIGAN PURDUE UNIVERSITY IOWA STATE UNIVERSITY

[About Us](#) | [Terms of Service](#) | [Contact Us](#)

NSDL Materials Digital Library Pathway: Targeting Undergraduates & Above

<http://matdl.org/matdlwiki>

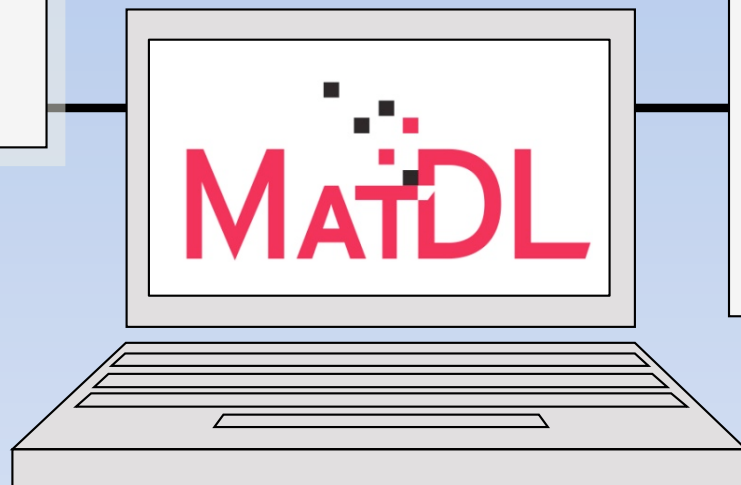
MatDL Expert Wiki

- Electrochemical
- Soft Matter Wiki

<http://matdl.org/virtuallabs>

Virtual Labs

- Intro to Solid State Chem (MIT)
- Intro to BioPhysics & 7 Ideas (KSU)
- General & Analytical Chemistry I (UW-M)
- Modern Chemistry (CMU)



Research & Ed Code Development

- Matforge
- E.g., NIST FiPy
- Ed: RPI

<http://matforge.org>

Stewardship

- MatDL Repository

<http://matdl.org>

Materials Failure Cases

- 12 institutions led by CSU

<http://matdl.org/failurecases>



Preview Online MatForge Demo

- MatForge
 - **MBuilder**: simulates 3D polycrystalline materials
 - **FiPy** & Teaching Resources

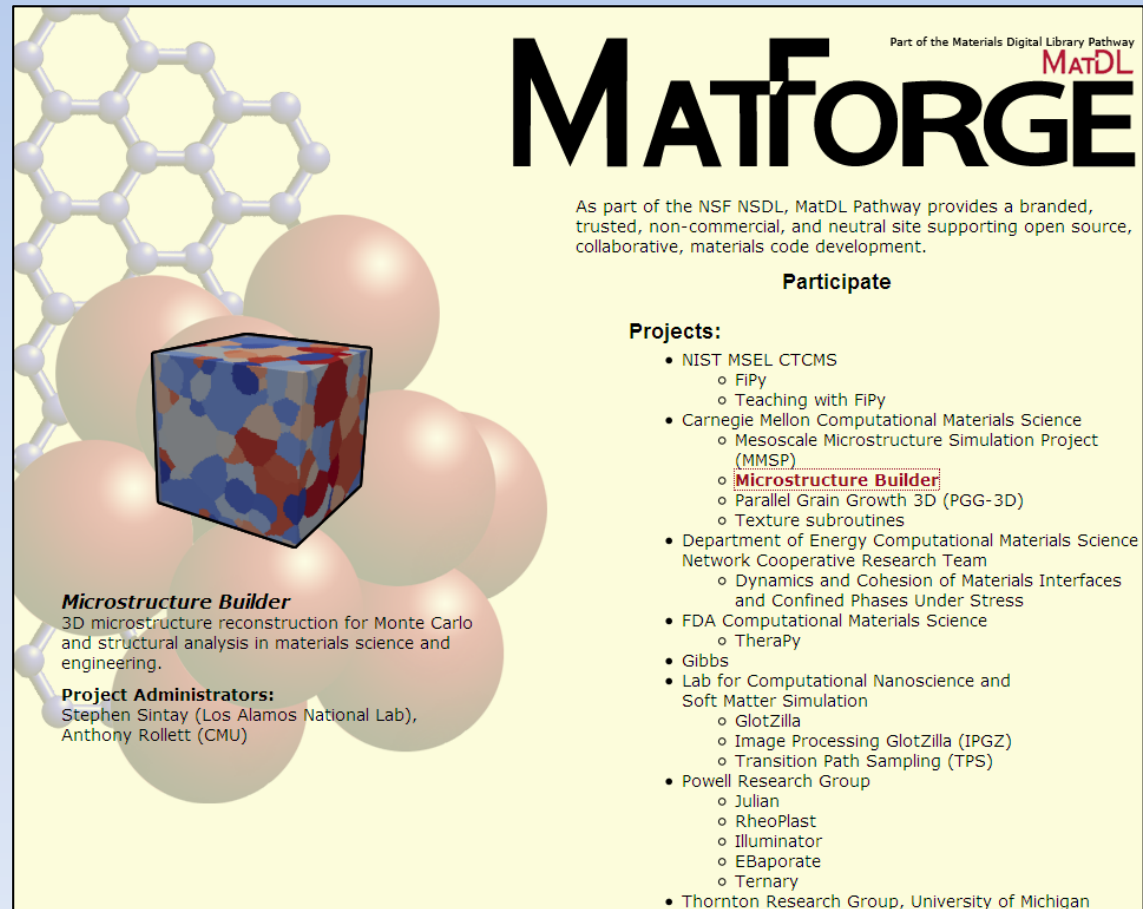
- Script Service
 - Carburization of Steel
 - Linked to teaching resources

Computational Community Tools

MatForge.org — Collaborative computational MS code development

Customizable workspace for research and education by:

- Hosting:
 - Develop on your machines, tools & timetable
 - Choose among various versioning control systems
 - Subversion, Git, Mercurial, Bazaar
 - Check in your code changes
 - Roll your code back
 - Daily backup
 - Communicate over encrypted channels
- Mirroring



Part of the Materials Digital Library Pathway
MATFORGE

As part of the NSF NSDL, MatDL Pathway provides a branded, trusted, non-commercial, and neutral site supporting open source, collaborative, materials code development.

Participate

Projects:

- NIST MSEL CTCMS
 - FiPy
 - Teaching with FiPy
- Carnegie Mellon Computational Materials Science
 - Mesoscale Microstructure Simulation Project (MMSP)
 - **Microstructure Builder**
 - Parallel Grain Growth 3D (PGG-3D)
 - Texture subroutines
- Department of Energy Computational Materials Science Network Cooperative Research Team
 - Dynamics and Cohesion of Materials Interfaces and Confined Phases Under Stress
- FDA Computational Materials Science
 - TheraPy
- Gibbs
- Lab for Computational Nanoscience and Soft Matter Simulation
 - GlotZilla
 - Image Processing GlotZilla (IPGZ)
 - Transition Path Sampling (TPS)
- Powell Research Group
 - Julian
 - RheoPlast
 - Illuminator
 - EBaporate
 - Ternary
- Thornton Research Group, University of Michigan

Microstructure Builder
3D microstructure reconstruction for Monte Carlo and structural analysis in materials science and engineering.

Project Administrators:
Stephen Sintay (Los Alamos National Lab),
Anthony Rollett (CMU)

Mbuilder: CMU & Los Alamos

The screenshot shows a web browser window with the URL matforge.org/mbuilder. The page features a navigation bar with links for Home, Projects, and Help, and a search box. The main content area is titled "Microstructure Builder" and includes a sub-navigation menu with links for Overview, Activity, Issues, News, Documents, Wiki, Files, and Repository. The "Overview" section contains two paragraphs: the first describes Mbuilder as a strategy for constructing simulated 3D polycrystalline materials from orthogonal images or 3D datasets, and the second states that the resulting voxel structures can be used for Monte Carlo simulations or converted to mesh structures for FE analysis. The "History of MBuilder" section provides background on the project's origin as a collaboration between David Saylor at CMU and Joe Fridy at Alcoa, supported by the NSF-funded MIMP project. The "Download" section offers a link to the source code archive. The "Compile Source" section lists the required software: Boost, CMake, C/C++/Fortran compilers, and Paraview. A sidebar on the right contains a "Wiki" section with links to the start page and indexes.

← → ↻ ↑ matforge.org/mbuilder ☆

For quick access, place your bookmarks here on the bookmarks bar. [Import bookmarks now...](#) Other bookmark

Home Projects Help Sign in

Search:

Microstructure Builder

Overview Activity Issues News Documents **Wiki** Files Repository

Microstructure Builder (MBuilder) ← History

Overview

Microstructure Builder or **MBuilder** is a strategy to construct simulated 3D polycrystalline materials. The input is typically grain size and shape data as obtained from orthogonal images (optical or SEM) or 3D datasets. The output is a 3D voxel structure that matches the size and shape statistics provided at input.

The voxel structures can be used directly as input to Monte Carlo simulations or can be converted to mesh structures for use in FE structural analysis.

History of MBuilder

Microstructure Builder started as a collaboration between David Saylor at Carnegie Mellon University (CMU) and Joe Fridy at the Alcoa Technical Center, with help from Tony Rollett, Bassem El-Dasher and Kyung-Jun Kung (all at CMU). It was supported by the Mesoscale Interface Mapping Project or MIMP under the NSF-supported Materials Research Science and Engineering Center at CMU (mimp.materials.cmu.edu). Various individuals have contributed to mbuilder over the years, including Chris Roberts, Abhijit Brahme, Sukbin Lee and Steve Sintay. Programs that have supported it include the Computational Materials Science Network (CMSN), DARPA under the SIPS program and the Commonwealth of Pennsylvania.

Download

Get the MBuilder source code archive
[[http://matforge.org/redmine/projects/mbuilder/repository/revisions/c85d2c6ff39ba7629dfba310950811aa28b3ab01/entry/mbuilder_1Feb11.tar.gz]]

Compile Source

MBuilder is compatible with Mac OSX, Linux/Unix, and Cygwin (for PC)

Need:

- Boost
- CMake
- C, C++, Fortran (77,95) compilers
- Paraview (for visualization)

Additional tools

Wiki

- [Start page](#)
- [Index by title](#)
- [Index by date](#)

MATFORGE Part of the Materials Pathway
MATDL

FiPy: NIST



FiPy: A Finite Volume PDE Solver Using Python

FiPy Home

[Login](#) | [Help/Guide](#) | [About Trac](#) | [Matforge Home](#) | [Register](#) | [Participate](#) | [Preferences](#)

Wiki	Timeline	Roadmap	Browse Source	View Tickets	New Ticket	Search	Downloader	Blog
----------------------	--------------------------	-------------------------	-------------------------------	------------------------------	----------------------------	------------------------	----------------------------	----------------------

[Start Page](#) | [Index](#) | [History](#) | [Last Change](#)

Overview

FiPy is an object oriented, partial differential equation (PDE) solver, written in [Python](#), based on a standard finite volume approach. The framework has been developed in the [Metallurgy Division](#) and Center for Theoretical and Computational Materials Science [\(CTCMS\)](#), in the Materials Science and Engineering Laboratory [\(MSEL\)](#) at the National Institute of Standards and Technology [\(NIST\)](#).

The solution of coupled sets of PDEs is ubiquitous to the numerical simulation of science problems. Numerous PDE solvers exist, using a variety of languages and numerical approaches. Many are proprietary, expensive and difficult to customize. As a result, scientists spend considerable resources repeatedly developing limited tools for specific problems. Our approach, combining the finite volume method and [Python](#), provides a tool that is extensible, powerful and freely available. A significant advantage to [Python](#) is the existing suite of tools for array calculations, sparse matrices and data rendering.

The **FiPy** framework includes terms for transient diffusion, convection and standard sources, enabling the solution of arbitrary combinations of coupled elliptic, hyperbolic and parabolic PDEs. Currently implemented models include phase field treatments of polycrystalline, dendritic, and electrochemical phase transformations as well as a level set treatment of the electrodeposition process .

The primary homepage for **FiPy** is at <http://www.ctcms.nist.gov/fipy>.

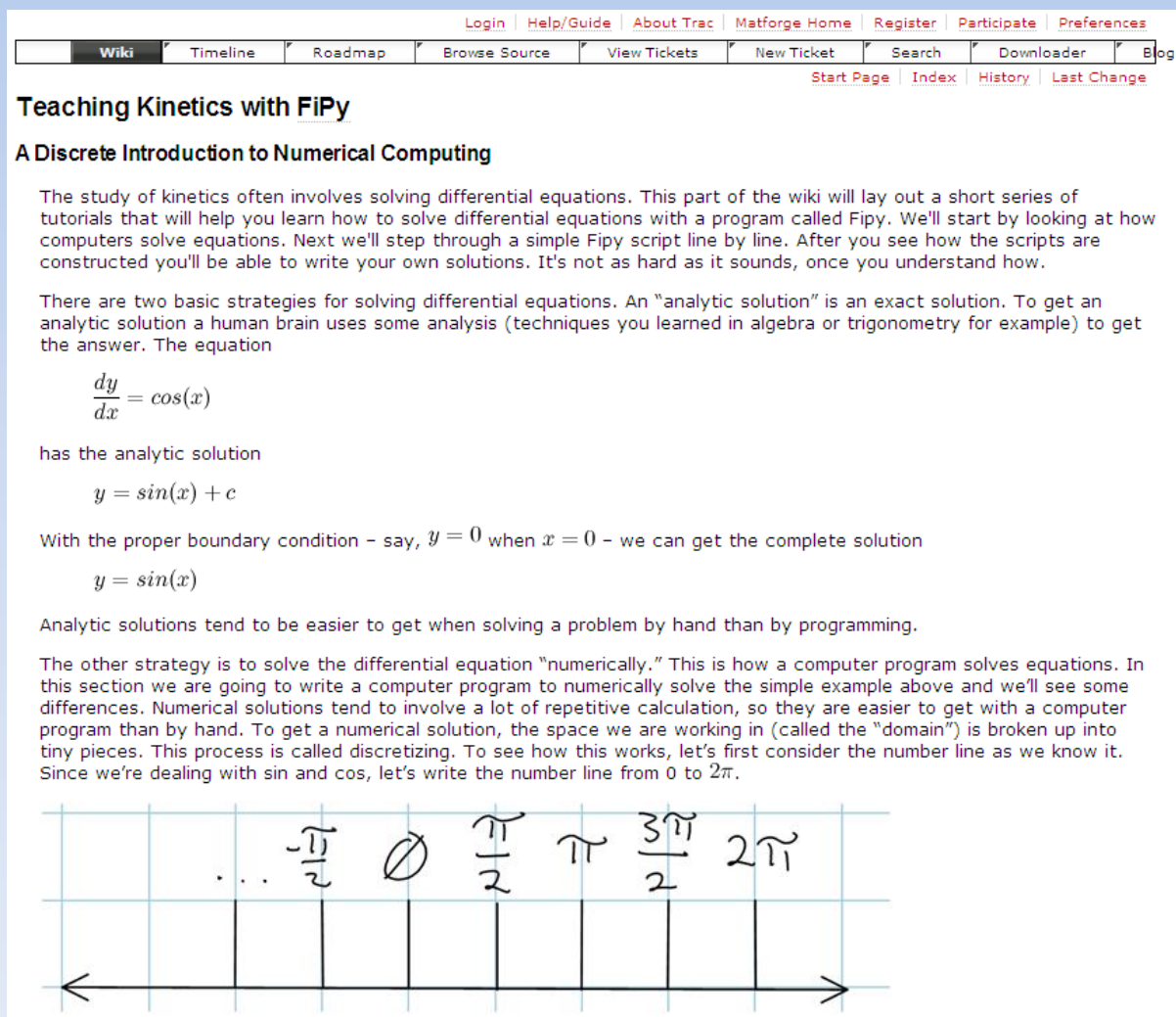
Other wiki pages:

- Examples provided by **FiPy** users:
 - [CookBook/BurgersEquation](#)
 - [CookBook/HeatTransfer](#)
- A wiki page about using **FiPy** in the classroom has been started at [TeachingWithFiPy](#)

MatForge & Teaching: FiPy workbook, Daniel Lewis, RPI

Exercises in 3 stages:

- Introduction to computational tools
- Kinetics basics
- Learn to write their own code to solve kinetics problems



The screenshot shows a Trac wiki page with a navigation bar at the top containing links for Login, Help/Guide, About Trac, Matforge Home, Register, Participate, and Preferences. Below the navigation bar is a secondary bar with links for Wiki, Timeline, Roadmap, Browse Source, View Tickets, New Ticket, Search, Downloader, and Blog. The main content area is titled "Teaching Kinetics with FiPy" and includes a sub-section "A Discrete Introduction to Numerical Computing". The text explains that the study of kinetics often involves solving differential equations and introduces the FiPy program. It provides an example of a differential equation $\frac{dy}{dx} = \cos(x)$ and its analytic solution $y = \sin(x) + c$. It also discusses numerical solutions and discretization, showing a number line from 0 to 2π with tick marks at $-\frac{\pi}{2}$, 0 , $\frac{\pi}{2}$, π , $\frac{3\pi}{2}$, and 2π .

Wiki | Timeline | Roadmap | Browse Source | View Tickets | New Ticket | Search | Downloader | Blog

Start Page | Index | History | Last Change

Teaching Kinetics with FiPy

A Discrete Introduction to Numerical Computing

The study of kinetics often involves solving differential equations. This part of the wiki will lay out a short series of tutorials that will help you learn how to solve differential equations with a program called FiPy. We'll start by looking at how computers solve equations. Next we'll step through a simple FiPy script line by line. After you see how the scripts are constructed you'll be able to write your own solutions. It's not as hard as it sounds, once you understand how.

There are two basic strategies for solving differential equations. An "analytic solution" is an exact solution. To get an analytic solution a human brain uses some analysis (techniques you learned in algebra or trigonometry for example) to get the answer. The equation

$$\frac{dy}{dx} = \cos(x)$$

has the analytic solution

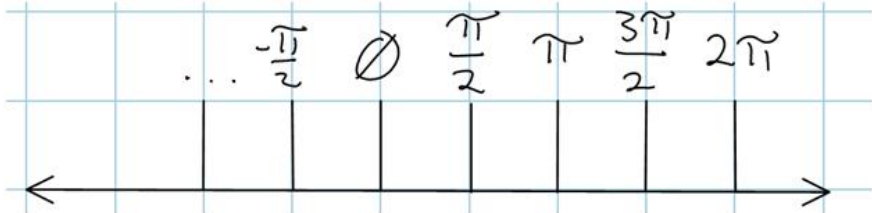
$$y = \sin(x) + c$$

With the proper boundary condition - say, $y = 0$ when $x = 0$ - we can get the complete solution

$$y = \sin(x)$$

Analytic solutions tend to be easier to get when solving a problem by hand than by programming.

The other strategy is to solve the differential equation "numerically." This is how a computer program solves equations. In this section we are going to write a computer program to numerically solve the simple example above and we'll see some differences. Numerical solutions tend to involve a lot of repetitive calculation, so they are easier to get with a computer program than by hand. To get a numerical solution, the space we are working in (called the "domain") is broken up into tiny pieces. This process is called discretizing. To see how this works, let's first consider the number line as we know it. Since we're dealing with sin and cos, let's write the number line from 0 to 2π .



New Script Service

Scripts/Examples/Modules:

- effective and efficient way to help bring modeling/simulation methods and research codes into the classroom
- increase functionality and rapid development of community-based code efforts
- lower barriers for participation in community-based code efforts

Online MatForge Demo

- MatForge.org
 - **MBuilder**: simulates 3D polycrystalline materials
 - **FiPy** & Teaching Resources
- Script Service
 - Carburization of Steel
 - Linked to teaching resources

Concluding Remarks

- Integrate CME modules into undergraduate courses
 - compelling/integral/advantageous to students
 - preparing next generation
- Computational mat'l research & education
 - bring together significant research code and vetted teaching resources
 - Take part in consortial academe/gov't/industry materials community with a sustainability plan
- MatDL: contribute to sharing community resources

Thank you

MatDL

<http://matdl.org>

MatForge

<http://matforge.org>

Laura Bartolo

lbartolo@kent.edu

This work is supported by the National Science Foundation 0532831, 0632726, 0817432 , 0919487 & 1115273. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

