02612
Constrained Optimization

Optimization Software

Software

- Gurobi (solvers for: LP, QP, QCP,MILP,MIQP,MIQCP)
- MOSEK (solvers for: LP, QP, SOCP, SDP, MIP)
- CVX (interface to solvers)
  - SeDuMi
  - SDPT3
- IPOPT (solver for NLP)
  - OPTI Toolbox
  - COIN-OR
- Matlab Optimization Toolbox
  - linprog
  - quadprog
  - Fmincon
- CasADi

Gurobi Optimizer

State of the Art Mathematical Programming Solver

The Gurobi Optimizer is the engine used by over 1500 companies in over two dozen industries to turn data into smarter decisions. It allows users to state their toughest business problems as mathematical models, and then automatically considers billions or even trillions of possible solutions to find the best one. Our solver can be used as both a decision-making assistant, to help guide the choices of a skilled expert, or as a fully automated tool to make decisions with no human intervention.

Gurobi has been used to produce measurable improvements in a wide range of high-value business functions, including production, distribution, purchasing, finance, capital investment and human resources. Gurobi has proven itself to be both robust and scalable, and is capable of solving problems involving millions of decision variables. The power of the library is backed by a broad range of intuitive interfaces that make it easy for new users to get up and running quickly, and the best technical support in the industry for when questions come up.

Recognized as the state-of-the-art solver for mathematical programming, Gurobi was designed from the ground up to exploit modern architectures and multi-core processors, using the latest implementations of the latest algorithms. This allows users more flexibility in how they model their problems, and increases their ability to add more complexity to their model both now and in the future, in order to better represent the real-world problems they are solving.

Beyond core performance, we've added a broad range of additional features to help users build and solve better models in less time. You can learn more about these and other features on our Features and Benefits page.

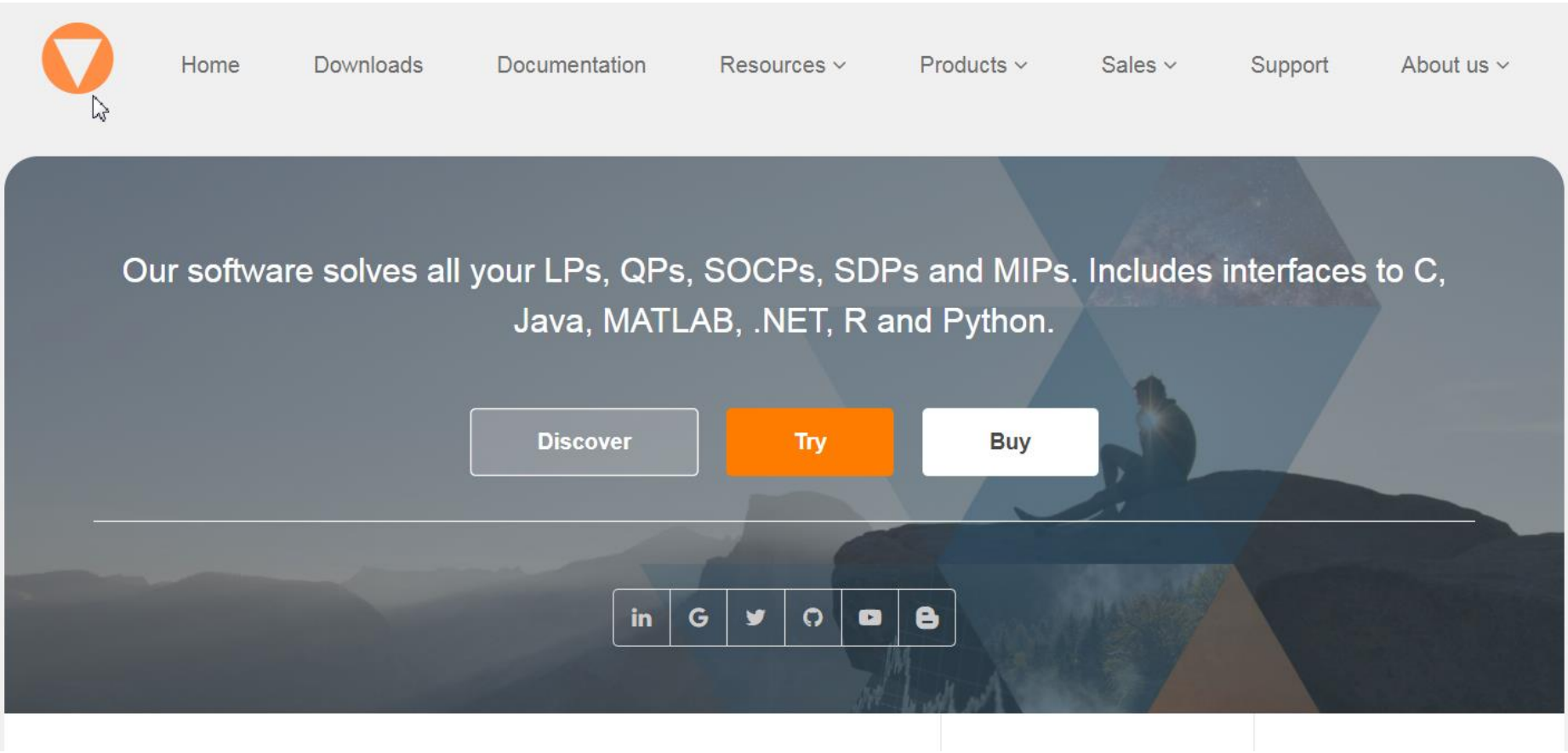
Solves all major problem types

The Gurobi Optimizer is a state-of-the-art solver for mathematical programming. The solvers in the Gurobi Optimizer were designed from the ground up to exploit modern architectures and multi-core processors, using the most advanced implementations of the latest algorithms. It includes the following solvers:

- linear programming solver (LP)
- mixed-integer linear programming solver (MILP)
- mixed-integer quadratic programming solver (MIQP)
- quadratic programming solver (QP)
- quadratically constrained programming solver (QCP)
- mixed-integer quadratically constrained programming solver (MIQCP)

http://www.gurobi.com

MOSEK



http://www.mosek.com

CVX RESEARCH

CVX TFOCS About us News CVX Forum

Software for Disciplined Convex Programming

CVX: a Matlab-based convex modeling framework

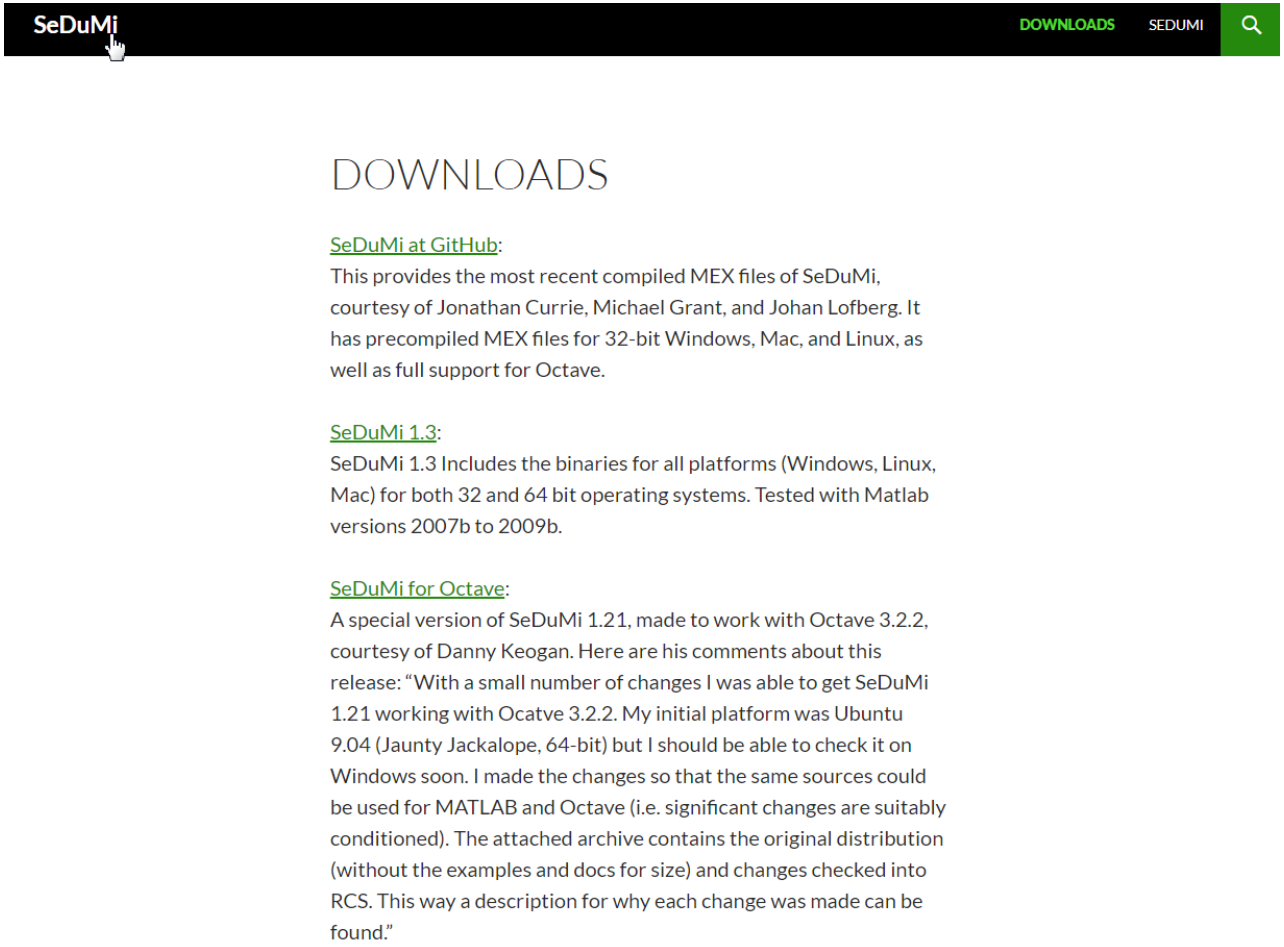
CVX is a popular modeling framework for disciplined convex programming that CVX turns Matlab into a modeling language, allowing constraints and objectives to be specified using standard Matlab expression syntax. CVX is a powerful tool for the rapid prototyping of models and algorithms incorporating convex optimization. Version 2.0, recently released in beta form, includes support for *mixed-integer* disciplined convex programs (MIDCPs). CVX is free for use in academic and commercial applications when paired with one of the included free solvers, SeDuMi or SDPT3. A *CVX Professional* license unlocks the ability to use CVX with commercial solvers Gurobi and Mosek. Academic users can obtain a CVX Professional license at no charge; commercial users can purchase licenses directly from CVX Research.

TFOCS: Templates for First-Order Conic Solvers

TFOCS (pronounced *tee-fox*) provides a set of Matlab templates, or building blocks, that can be used to construct efficient, customized solvers for a variety of convex models, including in particular those employed in sparse recovery applications. TFOCS can scale to larger model sizes than CVX. However, it is not a modeling framework—it requires all models to be manually converted to one of its standard forms. Therefore, it requires more expertise to use than CVX. TFOCS was conceived and written by Stephen Becker, Emmanuel J. Candès and Michael Grant, and is jointly owned by CVX Research and Caltech. It has been made freely available for both academic and commercial use under a BSD 3-Clause license.

http://www.cvxr.com

SeDuMi



http://sedumi.ie.lehigh.edu/?page_id=58

SDPT3

SDPT3 version 4.0 -- a MATLAB software for semidefinite-quadratic-linear programming

Kim-Chuan Toh , Michael J. Todd, and Reha H. Tutuncu

The last major update on the software was in Feb 2009. It implemented an infeasible path-following algorithm (sqlp.m) for solving SQLP -- conic optimization problems involving semidefinite, second-order and linear cone constraints. It also has a path-following algorithm (HSDsqlp.m) for solving a 3-parameter homogeneous self-dual reformulation of SQLP. Note: though this software is fairly well tested, but minor refinement or fix may still be needed from time to time.
New features that SDPT3 can now handle:
** free variables;
** determinant maximization problems;
** SDP with complex data;
** Matlab 7.3 on 64-bit machine;
** 3-parameter homogeneous self-dual model of SQLP (in HSDsqlp.m);

Citation:

- K.C. Toh, M.J. Todd, and R.H. Tutuncu, SDPT3 --- a Matlab software package for semidefinite programming, Optimization Methods and Software, 11 (1999), pp. 545--581.
- R.H Tutuncu, K.C. Toh, and M.J. Todd, Solving semidefinite-quadratic-linear programs using SDPT3, Mathematical Programming Ser. B, 95 (2003), pp. 189--217.

- Copyright: This version of SDPT3 is distributed under the GNU General Public License 2.0. For commercial applications that may be incompatible with this license, please contact the authors to discuss alternatives.
- SDPT3 is currently used as one of the main computational engines in optimization modeling languages such as CVX and YALMIP.
- **Download SDPT3-4.0.zip**
  Please read. Welcome to SDPT3-4.0! The software is built for MATLAB version 7.4 or later releases, it may not work for earlier versions.
  The software requires a few Mex files for execution. You can generate the Mex files as follows:
  - Firstly, unpack the software:
    unzip SDPT3-4.0.zip;
  - Run Matlab in the directory SDPT3-4.0
  - In Matlab command window, type:
    >> Installmex(1)
  - After that, to see whether you have installed SDPT3 correctly, type:
    >> startup
    >> sqlpdemo
  - By now, SDPT3 is ready for you to use.
- **User's guide** (pdf) (Draft)

- The following example shows how SDPT3 call a data file that is stored in SDPA format:

```
>> [blk,At,C,b] = read_sdpa('/sdplib/theta3.dat-s');
>> [obj,X,y,Z] = sdpt3(blk,At,C,b);
```

http://www.math.nus.edu.sg/~mattohkc/sdpt3.html

https://github.com/sqlp/sdpt3

http://www.math.cmu.edu/~reha/sdpt3.html

OPTI Toolbox

A free MATLAB Toolbox for Optimization

OPTI Toolbox Wiki

If you are interested in optimization, use MATLAB and like free stuff, OPTI could be for you. See the **What Is OPTI** section for details on solving linear, nonlinear, continuous and discrete optimization problems using MATLAB!

To get right into it, jump to the **downloads** page!

OPTI Toolbox v2.27 Released

written by jonny on july 10, 2017, at 08:54 pm

Version 2.27 has just been released with minor updates. Changes include:

- Updated installer to download the MEX files automatically from GitHub as a single zipped directory.
- For users pre-R2014b, detailed instructions on how to manually download the MEX files is now provided.

Apologies for the confusion while I have been sorting out the installer. Let me know if there are any problems!

OPTI Toolbox v2.25 Released

written by jonny on june 24, 2017, at 03:19 pm

Version 2.25 has just been released with minor updates. Changes include:

- Removed HTML documentation (problems with R2016b+ help index)
- Updated Visual Studio Builder for VS2017
- Rebuilt all solvers with VS2017
- Rebuilt all solvers against MKL v2017.0 R3
- Updated NOMAD to v3.8.1
- Updated IPOPT to v3.12.7
- Updated SCIP to v4.0.0

HOME

Return to Inverse Problem

SEARCH THIS WIKI

Go

OPTI TOOLBOX

Home Page
What Is OPTI?
Solvers
Utilities
Examples
Download OPTI
License
FAQ
Question & Answer Forum

http://www.inverseproblem.co.nz/OPTI/

OPEN SOURCE FOR THE OPERATIONS RESEARCH COMMUNITY

WHY OPEN SOURCE?

The **Open Source Initiative** explains it well. When people can read, redistribute, and modify the source code, software evolves. People improve it, people adapt it, people fix bugs. The results of open-source development have been remarkable. Community-based efforts to develop software under open-source licenses have produced high-quality, high-performance code on which much of the Internet is run.

WHY FOR OR?

Without open source implementations of existing algorithms, testing new ideas built on existing ones typically requires the time-consuming and error-prone process of re-implementing (and re-debugging and re-testing) the original algorithm. If the original algorithm were publicly available in a community repository, imagine the productivity gains from software reuse! Science evolves when previous results can be easily replicated.

OUR INITIATIVE

The COIN-OR project is managed by the COIN-OR Foundation, Inc., a non-profit educational foundation. We are building an open-source community for operations research software in order to speed development and deployment of models, algorithms, and cutting-edge computational research, as well as provide a forum for peer review of software similar to that provided by archival journals for theoretical research.

http://www.coin-or.org

IPOPT

Welcome to the Ipopt home page

Note that these project webpages are based on Wiki, which allows webusers to modify the content to correct typos, add information, or share their experience and tips with other users. You are welcome to contribute to these project webpages. To edit these pages or submit a ticket you must first register and login.

Introduction

Ipopt (**I**nterior **P**oint **OPT**imizer, pronounced eye-pea-Opt) is a software package for large-scale nonlinear optimization. It is designed to find (local) solutions of mathematical optimization problems of the from

```
   min     f(x)
x in R^n

s.t.       g_L <= g(x) <= g_U
           x_L <=  x   <= x_U
```

where `f(x): R^n --> R` is the objective function, and `g(x): R^n --> R^m` are the constraint functions. The vectors `g_L` and `g_U` denote the lower and upper bounds on the constraints, and the vectors `x_L` and `x_U` are the bounds on the variables `x`. The functions `f(x)` and `g(x)` can be nonlinear and nonconvex, but should be twice continuously differentiable. Note that equality constraints can be formulated in the above formulation by setting the corresponding components of `g_L` and `g_U` to the same value.

Ipopt is part of the COIN-OR Initiative.

Background

Ipopt is written in C++ and is released as open source code under the Eclipse Public License (EPL). It is available from the COIN-OR initiative. The code has been written by Andreas Wächter and Carl Laird. The COIN-OR project managers for Ipopt are Andreas Wächter und Stefan Vigerske. For a list of **all contributors**, see the AUTHORS file.

The C++ version has first been released on Aug 26, 2005 as version 3.0.0. The previously released pre-3.0 Fortran version is no longer maintained.

The Ipopt distribution can be used to generate a library that can be linked to one's own C++, C, Fortran, or Java code, as well as a solver executable for the AMPL modeling environment. The package includes interfaces to CUTEr optimization testing environment, as well as the MATLAB and R programming environments. IPOPT can be used on Linux/UNIX, Mac OS X and Windows platforms.

As open source software, the source code for Ipopt is provided without charge. You are free to use it, also for commercial purposes. You are also free to modify the source code (with the restriction that you need to make your changes public if you decide to distribute your version in any way, e.g. as an executable); for details see the EPL license. And we are certainly very keen on feedback from users, including contributions!

In order to compile Ipopt, certain third party code is required (such as some linear algebra routines). Those are available under different conditions/licenses.

If you want to learn more about Ipopt, you can find references in the bibliography of the documentation and the "Papers about Ipopt" page.

For information on projects that use Ipopt, refer to the Success Stories page.

http://projects.coin-or.org/Ipopt

Matlab Interface to IPOPT

- https://projects.coin-or.org/Ipopt/wiki/MatlabInterface

- Mac / Unix:
  - https://github.com/ebertolazzi/mexIPOPT
  - https://projects.coin-or.org/Ipopt/wiki/Ipopt_on_Mac_OS_X

- The Matlab Interface
  - https://www.coin-or.org/Ipopt/documentation/node28.html

CasADi – www.casadi.org

CasADi
– installation and getting started

Install CasADi? Pick a version from below...

... or try out CasADi live in your browser (Python, Octave/Matlab).

CasADi v3.5.5 released September 5, 2020

Install | Release notes

Grab a binary from the table (for MATLAB, use the newest compatible version below):

	Windows	Linux	Mac (High Sierra or above)
Matlab	R2016a or later, R2014b, R2014a, R2013a or R2013b	R2014b or later, R2014a	R2015a or later, R2014b, R2014a
Octave	4.4.1 (32bit / 64bit), 4.4.0 (32bit / 64bit), 5.1.0 (32bit / 64bit), 5.2.0 (32bit / 64bit), 6.1.0 (32bit / 64bit)	4.4.1, 5.1.0, 5.2.0, 6.1.0	5.2.0, 6.1.0(Mojave or higher)
Python	Py27 (32bit / 64bit), Py35 (32bit / 64bit), Py36 (32bit / 64bit), Py37 (32bit / 64bit), Py38 (32bit / 64bit) Py39 (32bit / 64bit)	Py27, Py35, Py36, Py37, Py38, Py39	Py27, Py35, Py36, Py37, Py38, Py39
	or just `pip install casadi` (needs `pip -V` >=8.1)		

(*) Check your Python console if you need 32bit or 64bit - bitness should be printed at startup.

Unzip in your home directory and adapt the path:

Matlab/Octave

```
addpath('<yourpath>/casadi-matlabR2014a-v3.5.5')
import casadi.*
x = MX.sym('x')
disp(jacobian(sin(x),x))
```

Python

```
from sys import path
path.append(r"<yourpath>/casadi-py27-v3.5.5")
from casadi import *
x = MX.sym("x")
print(jacobian(sin(x),x))
```

Get started with the example pack.

This tag on GitHub

CasADi examples

ODE and sensitivities

```
% Casadi example 1

import casadi.*


x = MX.sym('x',2); % Two states

% Expression for ODE right-hand side
z = 1-x(2)^2;
rhs = [z*x(1)-x(2); x(1)];

ode = struct;    % ODE declaration
ode.x   = x;     % states
ode.ode = rhs;   % right-hand side

% Construct a Function that integrates over 4s
F = integrator('F','cvodes',ode,struct('tf',4));

% Start from x=[0;1]
res = F('x0',[0;1]);

disp(res.xf)

% Sensitivity wrt initial statEe
res = F('x0',x);
S = Function('S',{x},{jacobian(res.xf,x)});

disp(S([0;1]))
```

Optimization problem

```
%example2CasADi


import casadi.*


% Symbols/expressions
x = MX.sym('x');
y = MX.sym('y');
z = MX.sym('z');
f = x^2+100*z^2;
g = z+(1-x)^2-y;

nlp = struct;              % NLP declaration
nlp.x = [x;y;z];           % decision vars
nlp.f = f;                 % objective
nlp.g = g;                 % constraints

% Create solver instance
F = nlpsol('F','ipopt',nlp);

% Solve the problem using a guess
res = F('x0',[2.5 3.0 0.75],'ubg',0,'lbg',0)
```

Optimal control problem

```
    % example 3 CasADi

import casadi.*


x = MX.sym('x',2); % Two states
p = MX.sym('p');   % Free parameter

% Expression for ODE right-hand side
z = 1-x(2)^2;
rhs = [z*x(1)-x(2)+2*tanh(p);x(1)];

% ODE declaration with free parameter
ode = struct('x',x,'p',p,'ode',rhs);

% Construct a Function that integrates over 1s
F = integrator('F','cvodes',ode,struct('tf',1));

% Control vector
u = MX.sym('u',5,1);

x = [0;1]; % Initial state
for k=1:5
  % Integrate 1s forward in time:
  % call integrator symbolically
  res = F('x0',x,'p',u(k));
  x = res.xf;
end

% NLP declaration
nlp = struct('x',u,'f',dot(u,u),'g',x);

% Solve using IPOPT
solver = nlpsol('solver','ipopt',nlp);
res = solver('x0',0.2,'lbg',0,'ubg',0);

%plot(full(res.x))
figure(1)
stairs(0:4,full(res.x),'linewidth',2)
```