Call #1: December 5, 2021

Agenda

- Introductions
- CVXPY is joining NumFOCUS! What that means for us. [Steven]
    - Discuss logo
- Non-code initiatives [Akshay]
    - Solver interface owners
    - Enhancing CVXPY examples and tutorials
    - Community-driven blog?
- Adopting a release schedule and semantic versioning. [Riley]
    - Comments on The CVXPY roadmap
- Code generation demo [Max + Bartolomeo]
- Open discussion!
    - Free-form open discussion. Example topics below.
    - What tasks should we prioritize?
        - Problem serialization? (1, 2)
        - Faster compilation?
        - New problem classes (such as SCS 3.0 standard form)?
        - MIP modeling tools?
        - Something else?
    - Pain points: how can we make CVXPY users (or developers) more productive?
    - Share how you use CVXPY
    - Thoughts on how to grow the CVXPY community?
    - Potential developer workshop

"Minutes" (really, miscellaneous comments, notes, and action items)

- Attendees: Steven, Bartolomeo, Akshay, Riley, Phillip, Phillipe, Max, MIchael, and Adi.
- Improving examples for new users
    - Have notebooks runnable on Binder or Google Colab
    - Regularly run example notebooks (they're often broken)
    - Have notebooks that don't even assume people know what an LP is.
- Logo
    - Steven will share NumFOCUS drafts on Discord.
    - Adi designed the OpenCV logo; happy to be a part of CVXPY logo design
    - Note: logo is *only for CVXPY*! Nothing else in "cvx" world.
- Community engagement

    - Bart suggests a blog where CVXPY users share their success stories.
        - Adi says his company could probably contribute a post.
    - Adi says it would be nice to have (local and/or online) meet-ups; organizes some around C++ development.
    - Gauging size of community
        - We don't know how many active users
        - We get 450k - 600k PyPI downloads / month
        - GitHub says CVXPY is used in 4k other repos.
        - Riley suggestion not from meeting: we should have user surveys. NumPy and LAPACK send out user surveys regularly. We can advertise for the surveys on Discord and Twitter.
- Canonicalization speed
    - Code gen makes a big difference for small problems
    - Adi's ad-hoc serialization helps amortize significant canonicalization cost (~30 seconds to ~2 minutes) over many calls to GUROBI (which runs in milliseconds).
    - Much of the inefficiency is due to poorly designed data structures in cvxcore.
- Improvements to cvxcore
    - Priority #1: problems in standard-form without any Parameter objects.
    - Cvxcore uses its own sparse tensor format. We could look at projects dedicated to sparse tensors (e.g., https://sparse.pydata.org/en/stable/index.html).
- Code Gen
    - Build models in Python, deploy them in C on embedded systems.
    - Changing a Parameter updates only the relevant parts of problem data (e.g., the relevant rows of the constraint matrix).
    - Within C, the pre-canonicalization (i.e., user-defined) Variables are usually just pointers to canonicalized representations of the Variable. Some exceptions for symmetric matrix variables.
    - Repo should be public soon.
- Semantic versioning
    - We're moving to semantic versioning. We'll support multiple minor versions at once. This helps projects that use CVXPY, helps motivate developers, and helps keep users engaged.
    - Big questions (discussion to be had on Discord)
        - Release schedule. Fixed interval (e.g., every 4 months) or a lower-bound on time-between-releases?
        - When to release 1.2?
        - How many minor versions to support at any given time?
    - See slides for more details
- Next call to be scheduled on Discord.

- Other remarks (If you don’t have write access to this document, use Google Doc’s comment feature by your name below).
    - Phillip
        - When someone opens a github issue for the first time, we can use github actions to automatically comment. For example, we can point them to the discord, emphasize that they should include a reproducible code snippet, etc.
        - We can give people the “issue maintainer” role, where they close old issues, follow up when necessary, and generally make sure issues are getting proper attention.
    - Phillipe
    - Max
    - Michael
    - Adi: has a fork of cvxpy with two patches to the Expression system that greatly improve modeling power. We could merge into a branch of cvxpy and explore it more ourselves.