Supporting Open Source Mathematical Software through Automatic Accurate Citations

Barry Smith
Argonne National Laboratory
Mandatory Citation

@techreport{knepley2013accurately,
    title = {Accurately Citing Software and Algorithms Used in Publications},
    author = {Matthew G Knepley and Jed Brown and Lois Curfman McInnes and Barry F. Smith},
    year = {2013},
    institution = {http://files.figshare.com/1187013/paper.pdf}
}
Dilemma: Open Source Numerical Software is Highly Valuable but Not Worth Spending Money On

- **Grants** - There is very little direct financial support for the development or maintenance of open source numerical software. Funding is usually in the context of
  - research on algorithms or
  - research on specific application areas

- **Careers** - Implementing and maintaining, even widely used, open source numerical software is rarely enough on its own to develop and maintain a scientific research career
  - Widely cited publications are the gold standard for career recognition and advancement, followed by invited presentations, awards, etc.

- Is developing/maintaining open source numerical software wasted effort?
Open Source Software is Often Not Cited

- Users may not know it matters,
  - For example, “PETSc is developed at Argonne National Laboratory, surely they have stacks of money to pay for it; its more important for me to cite that new algorithm by that young scientist,”
- May be near the bottom of the software stack, hidden from end user
  - “I cited MOOSE, I didn’t realize most of the work is done internally by libMESH, PETSc, and hypre”.
- Users may not know what to cite,
  - “I used some solver in SuperLU, not sure which one”
  - Software packages receive a variety of funding over the years, the proper citation credits the proper work.
Solution

The Software Should Tell you the Correct Citations based on the given computations

```
./ex19 -ksp_monitor -citations -pc_type hypre -pc_hypre_type boomeramg

./ex19 -citations -pc_type lu -pc_factor_mat_solver_package superlu

./ex19 -snes_type anderson -citations -snes_view

./ex19 -pc_type fieldsplit -snes_view -fieldsplit_Omega_pc_type hypre
         -fieldsplit_temperature_pc_type ml -citations
```
Implementation

PetscCitationsRegister(const char bibtexentry[], bool *set)

PetscErrorCode MatSolve_SuperLU_Private(Mat A, Vec b, Vec x) 
  Mat_SuperLU *lu = (Mat_SuperLU*)A->spptr;

... static PetscBool cite = PETSC_FALSE;
  PetscCitationsRegister("@article{superlu99,\n  author = {James W. Demmel and Stanley C. Eisenstat and \n            John R. Gilbert and Xiaoye S. Li and \n            Joseph W. H. Liu},\n  title = {A supernodal approach to sparse partial pivoting}, \n  journal = {SIAM J. Matrix Analysis and Applications}, \n  year = {1999}, \n  volume = {20}, \n  number = {3}, \n  pages = {720-755}\n}, &cite);
Generalization

CitationsRegister(const char bibtensexentry[], bool *set)
CitationsPrint(FILE *)

What about parallelism?

- Different libraries may be used on different nodes of the machine.
- Need to communicate information down to single “printer”

Alternatives?

- Provide URL to citation information instead of actual bibtex entry?
- Provide individual entries of bibtex item, not just single string?

Wrap up code so it can be included in many libraries but not conflict at link time?

- Weak symbols, separate library?
Off-Topic - Monitoring Long Running Simulations

@TechReport{ saws,
  Author = "Matt Otten and Jed Brown and Barry Smith",
  Title = "Scientific Application Web Server (SAWs) Users Manual",
  Institution = "Argonne National Laboratory",
  Year = 2013
  URL = "http://bitbucket.com/saws/saws"
}

- A tool to publish variables in long running simulations to the web. Library that turns your application into a webserver.
- Based on the standard RESTful software architecture; easily used from Javascript.

./ex5  -ts_view_pre saws  -stack_view saws -draw_save -draw_save_single_file
-x_virtual -ts_monitor_draw_solution -saws_root . -saws_local
http://petsc.no-ip.org:8080
SAWs API

- SAWs_Initialize();
- SAWs_Register(const char *variablepathandname, void* addr, int len, SAWs_Memory_type mtype, SAWs_Data_type dtype);
  - variablepathandname is of the form [/dir1[/dir2[/dir3]]]/
  - mtype determines if the client is allowed to change the variable (SAWs_WRITE) or only read it (SAWs_READ)
  - dtype is one of SAWs_CHAR, SAWs_BOOLEAN, SAWs_INT, SAWs_FLOAT, SAWs_DOUBLE, or SAWs_STRING
- SAWs_Delete(const char *variablepathandname)
- SAWs_Lock()
- SAWs_Unlock()
- SAWs_Selected(const char *variablepathandname, int *changed);
- SAWs_Finalize();
Sample Usage

- PetscSNPrintf(dir,1024,"/PETSc/Objects/%s/its",name);
- SAWs_Register,(dir,&ksp->its,1,SAWs_READ,SAWs_INT);

- PetscSNPrintf(dir,1024,"/PETSc/Objects/%s/res_hist",name);
- SAWs_Register,(dir,ksp->res_hist,10,SAWs_READ,SAWs_DOUBLE);