LINEAR ALGEBRA EDUCATION

The Sage Mathematical Software System

Jason Grout, Drake University, USA, jason.grout@drake.edu

Sage (`http://sagemath.org`) is a comprehensive free open-source mathematics software system. Anybody is welcome to freely download, modify, and redistribute copies of Sage. Alternatively, several free services exist to use Sage online, without needing to install anything. Sage can be used in teaching and researching many areas of mathematics. In this article, we will briefly discuss Sage's linear algebra capabilities, show how to use Sage through two online interfaces, and then discuss using Sage in teaching.

Sage and Linear Algebra

Rob Beezer, Robert Bradshaw, William Stein, and I recently wrote a chapter for the 2nd edition of Leslie Hogben's *Handbook of Linear Algebra*, introducing Sage's linear algebra capabilities. This chapter is also distributed under an open-source license (i.e., copying and redistribution is welcome), available from `http://artsci.drake.edu/grout/doku.php/ilas`. We highly recommend reading this more complete introduction to linear algebra in Sage.

Sage's core is a unified framework based on category theory for working with mathematical concepts. Sage can work with theoretical mathematical objects like rings, fields, vector spaces, linear transformations, and much more. Vectors and matrices in Sage know the rings in which their elements live (such as integers, rationals, finite fields, floating-point reals or complexes, or more exotic objects such as intervals, multivariate polynomials, or algebraic numbers). When a user asks Sage to do a computation, Sage automatically chooses appropriate routines for the object, often using fast standard packages included in Sage such as LAPACK, LinBox, ATLAS, IML, M4RI, etc. Users interact with Sage via Python, a very popular, easy-to-use, and powerful general-purpose programming language. Sage also includes popular open-source packages like GAP, R, Singular, and Pari, and makes it easy to use external packages from within Sage, like Octave (an open-source MATLAB clone).

In Sage, to create the vector $(1, 2, 3)$ over the rationals and assign it the name "v", type `v=vector(QQ, [1,2,3])` (the `QQ` tells Sage the vector is over the rationals). To create a matrix over the rationals and assign it the name "A", just specify the rows of the matrix like `A=matrix(QQ, [[1,2,3],[4,5,6],[7,8,9]])`. Vector and matrix entries are retrieved by giving indices in square brackets. Indices count from 0, as is common in many computer languages, so `A[0,0]` retrieves the upper left entry of the matrix A, and `v[2]` retrieves the third element of $\vec{v}$. To solve the system $A\vec{x} = \vec{v}$, do `A\v`. You can get information about an object or perform operations on an object by typing the object name, a period, and then a method name followed by parentheses. For example, the determinant and eigenvalues of A are found by doing `A.det()` and `A.eigenvalues()`, and the Euclidean norm of $\vec{v}$ is `v.norm()`. You can get an overview of what methods are available by typing an object's name, a period, and then pressing the tab key. Sage will then display a list of possible methods you could invoke. To get help about a method, type the method name followed by a question mark, and then press tab: `A.rref?`. To see the underlying source code for a method, append two questions marks and press tab: `A.rref??`.

As mentioned above, Sage understands theoretical mathematical objects like vector spaces and linear transformations. To get the (right) kernel of a matrix as a vector space and assign it the name "K", do `K=A.right_kernel()`. Among other things, you can ask for a basis for the vector space `K.basis()`, the dimension `K.dimension()`, or the coordinate vector of $(-2, 4, -2)$ relative to the basis `K.coordinate_vector((-2,4,-2))`. You can construct a vector space in many other ways, including raising a field to a power, `QQ^2`, or as the span of a list of vectors `span([v])`.

Sage also understands and can compute directly with linear transformations. You can construct a linear transformation, among other ways, by giving a matrix, like `T=linear_transformation(A,side='right')`. The `side='right'` is to tell Sage that vectors act on the right of the matrix (i.e., $A\vec{x}$). You can compute many things from linear transformations, like the image `T.image()` and the kernel `T.kernel()`. You can compose and restrict transformations to create new ones.

Sage also has many graphics capabilities: `plot(v, start=(1,2,3))+plot(A*v)` draws $\vec{v}$ and $A\vec{v}$ (with v starting at the point $(1, 2, 3)$) in an interactive 3d graph. Doing `matrix_plot(A)` visualizes the structure of a matrix A.

Sage Cell Server: `https://sagecell.sagemath.org`

There are many ways to use Sage, including downloading an app for Apple OS X or Linux, a virtual machine image for Windows, or using the Sage apps on iPhones, iPads, or Android devices. Two of the easiest ways to use Sage are via the online Sage Cell Server and Sage Cloud. Using Sage online provides the full capabilities of Sage with no installation.

goo.gl/VaGjqt

The Sage Cell Server at `https://sagecell.sagemath.org` provides the easiest way to do short computations in Sage: just type in the code and press the Evaluate button. You can also share a snippet of code by clicking the "Share" button, which reveals permalinks and a QR code. A QR code is a square-shaped 2D barcode, such as the one above, which

encodes a permalink. A user with a QR code scanner installed on their mobile device can take a picture of the barcode and automatically go to the permalink. I also included shortened versions of the permalinks under the QR codes in this article. I have fruitfully used permalinks extensively in pdf and online notes—students click on the link to get a snippet of Sage code that they can run and modify to check their work or explore a situation.

Sage cells can also be embedded into any webpage (click the "About" link in the upper right of the Sage cell server page for instructions). Many people embed live Sage cells into webpages for their homework assignments, blogs, and online notes. For example, Rob Beezer's open-source online linear algebra textbook uses Sage cells extensively to illustrate concepts throughout the book; see `http://linear.ups.edu/html/section-RREF.html` for an example (click on the Sage links). Since Sage provides interfaces to other programs, such as Octave (an open-source clone of MATLAB) and R (a popular statistics program), it is easy to embed examples and make permalinks using those languages as well (see the QR code to the right for an example).

goo.gl/UTrQt0

Sage Cloud: `https://cloud.sagemath.com`

William Stein, the lead developer of Sage, has been developing a new online interface to Sage, the Sage Cloud at `https://cloud.sagemath.com`. Currently in beta status, it is already a powerful computation and collaboration tool. Work is organized into projects which can be shared with others. Inside a project, you can create any number of files, folders, Sage worksheets, LaTeX documents, code libraries, and other resources. Real-time collaborative editing allows multiple people to edit and chat about the same document simultaneously over the web. The LaTeX editor features near real-time preview, forward and reverse search, and real-time collaboration. Also, it is easy to have Sage do computations or draw figures and have those automatically embedded into a LaTeX document using the SageTeX package (for example, after including the sagetex package, typing `\sageplot{plot(sin(x))}` in a TeX document inserts the plot of $\sin x$). A complete Linux terminal is also available from the browser to work within the project directory. Snapshots are automatically saved and backed up every few minutes to ensure work is never lost. William is rapidly adding new features, often within days of a user requesting them.

Interacts

Sage interacts make it easy to create online interfaces which include buttons, sliders, menus, etc. A visitor on the webpage (or a collaborator in the Sage Cloud) can adjust parameters and see the results immediately, without having to enter or change any Sage code. Interacts are especially effective when they are embedded into websites using the Sage cell server. The code can be hidden so the user just sees an "Evaluate" button, and clicking the button pops up the user interface. Examples of interacts can be found at `http://wiki.sagemath.org/interact` and `http://interact.sagemath.org`.

Creating interacts is straightforward. For example, suppose we want an exploration plotting $\vec{v}$ and $A\vec{v}$ for various unit vectors $\vec{v} \in \mathbb{R}^2$. We can do that with the code:

```
angle = pi/6
A = matrix([[1,-1],[2,1]])
v = vector([cos(angle), sin(angle)])
plot(v, color='red')+plot(A*v, color='blue')
```

goo.gl/Ek3A4X

To turn this into an interact, we enclose the code in a function (put in the `def` line below and indent the code), make the variables we want the user to change into inputs of the function (give a range for numeric variables, like `angle`), and put `@interact` in front of the entire thing. We also wrap any plots in the `show()` function:

```
@interact
def myfunction(angle=(0,2*pi), A=matrix([[1,-1],[2,1]])):
    v = vector([cos(angle), sin(angle)])
    show(plot(v, color='red')+plot(A*v, color='blue'))
```

goo.gl/HQCsZ8

Sage in Teaching

Sage is being used in many ways in teaching. Sage snippets and explorations are embedded into online textbooks, notes, and homework assignments. Permalinks and QR codes to Sage snippets in pdf documents and printed material make it easy for students to explore and check their work with a single click or picture. Teachers use SageTeX to author tests and write TeX documents that include Sage computations and graphics. The online homework system Webwork can use Sage calculations in creating homework problems. The Sage Cloud is being used as a collaborative environment for a teacher and students to author class notes, submit and grade homework assignments, and do computational projects.

The advantages to using Sage go beyond the immediate class. The Python language that students learn when interacting with Sage is widely used in many areas outside of mathematics, and is particularly prevalent in scientific computing.

Since Sage is open-source, students can dive as deep as they want into the code to understand how algorithms work. Many undergraduate and graduate students also write and contribute code to Sage—this enhances their understanding and gives them exposure to the wider professional community.

Learn More

There are many resources and opportunities to learn about Sage and interact with the community of developers and users. Sage Days and Sage Education Days workshops happen often and new users are welcome. Funding has usually been available to help faculty and students attend. See `http://wiki.sagemath.org/Workshops` for a listing of past and future workshops, along with videos of past talks and other resources (the Education Days at the bottom of the listing have resources specifically for teaching). Additionally, the sage-support mailing list at `https://groups.google.com/forum/#!forum/sage-support` and the Ask Sage website at `http://ask.sagemath.org` provide very helpful forums for new users.

ILAS NEWS

ILAS Member Elected to Executive of SIAM

D. Szyld

Professor Daniel Szyld (Temple University) has been elected by SIAM (*Society for Industrial and Applied Mathematics*) to a two-year term as Vice-President-at-Large, starting January 1, 2014.

Professor Szyld has been an active member of both ILAS and SIAM. A member of ILAS since its foundation, he was an ILAS board member from 2001–2004. He was instrumental in helping set up the *Electronic Journal of Linear Algebra* (ELA), and served as ELA's first managing editor from 1995–2003, as well as an associate editor from 1995–2001. He has been an advisory editor of ELA since 2001. He is currently serving on the editorial board of several other journals, including *Linear Algebra and its Applications*, *Numerical Linear Algebra with Applications*, and the *SIAM Journal on Matrix Analysis and Applications*. Professor Szyld's activities include membership on the organizing committees of a couple of ILAS conferences (1998 and 2008), and he has also been a program committee member for the SIAM Conference on Applied Linear Algebra in 2009, as well as chair of the Gene Golub Summer School (2010 – 2013) sponsored by SIAM. A member of SIAM since 1980, he chaired the SIAG-LA, the linear algebra subgroup of SIAM, from 2007–2009. Daniel notes that "the SIAG-LA has a good working relationship with ILAS" and he hopes to continue to support this positive relationship.

Nominations for 2013 ILAS Elections

Submitted by Stephen Kirkland, ILAS President

The Nominating Committee for the 2013 ILAS elections has completed its work. Nominated for a three year term, beginning March 1, 2014, as ILAS President are: André Ran (Netherlands) and Peter Šemrl (Slovenia).

Nominated for the two open three-year terms, beginning March 1, 2014, as "at-large" members of the ILAS Board of Directors are: Froilán Dopico (Spain), Julio Moro (Spain), Michael Overton (USA), and Eugene Tyrtyshnikov (Russia).

Many thanks to the Nominating Committee for their important service to ILAS: Shaun Fallat, Heike Faßbender, Roger Horn (chair), Yimin Wei, and Zdenek Strakoš. Thanks also to the nominees for agreeing to stand for election.

Voting in this election is electronic, using Votenet Solutions, with instructions sent to members by email.

ILAS Website Update

Please note that ILAS has a new website address: `http://www.ilasic.org/`. Please take the time to update your bookmarks and any links you may have. If you have any suggestions for the new website, please contact Sarah Carnochan Naqvi, the ILAS-IC and ILAS-Net Manager, at ilasic@uregina.ca.