Untangling Random Polygons and Other

By Charles F. Van Loan

The following is an introduction to the 2018 John von Neumann Lecture, to be presented at the upcoming SIAM Annual Meeting (AN18) in Portland, Ore., from July 9-13. Look for feature articles by other AN18 invited speakers introducing the topics of their talks in future issues.

The *polygon averaging problem* involves just about everything I like with respect to teaching and research in matrix computations: my favorite matrix decomposition (the singular value decomposition), my favorite matrix dimension ($N=2$), and my favorite structured matrix (the "upshift matrix"). Even better, it evolved from an assignment that I gave in a MATLAB-based CS1 introduction, a reminder of just how much there is to learn when working at that level.

Here's the problem: suppose we have a random polygon $\mathcal{P}_0$ with vertices $(x_1,y_1),\ldots,(x_n,y_n)$. Assume that $\mathcal{P}_0$ has centroid $(0,0)$, and $\|x\|_2=1$ and $\|y\|_2=1$. If we connect the midpoints of its edges we obtain a new polygon, also with centroid $(0,0)$. The "next" polygon $\mathcal{P}_1$ is obtained by scaling the x- and y-values so that $\|x\|_2=1$ and $\|y\|_2=1$. The process can obviously be repeated to produce a sequence of polygons $\{\mathcal{P}_k\}$. The assignment required students to plot a reasonable number of $\mathcal{P}_k$. They observed something truly amazing. No matter how "criss-crossy" the initial polygon $\mathcal{P}_0$, the $\mathcal{P}_k$ eventually "untangle" and their vertices head towards an ellipse with a 45-degree tilt (see Figure 1).

This astonished me as well, because I did not work out the solution before handing out the assignment. But that is precisely why I like teaching CS1 courses — they are both an integral part of STEM education and full of surprises. Teaching at the CS1 level allows me to build intuition for linear algebra. The one-dimensional array—a vector in MATLAB—typically marks the first time that a beginner STEM student sees n things as one thing. Yes, a polygon is defined by its n vertices, but a student must think at the vector level when writing the function

```
[xTilde,yTilde]=PolygonAve(x,y),
```

which produces a new, midpoint-connected polygon from the old one. Behind the scenes is a sparse matrix-vector product, e.g.,

$$\begin{pmatrix}(x_1+x_2)/2\\(x_2+x_3)/2\\(x_3+x_4)/2\\(x_4+x_1)/2\end{pmatrix}=\frac{1}{2}\begin{pmatrix}1&1&0&0\\0&1&1&0\\0&0&1&1\\1&0&0&1\end{pmatrix}\begin{pmatrix}x_1\\x_2\\x_3\\x_4\end{pmatrix}.$$

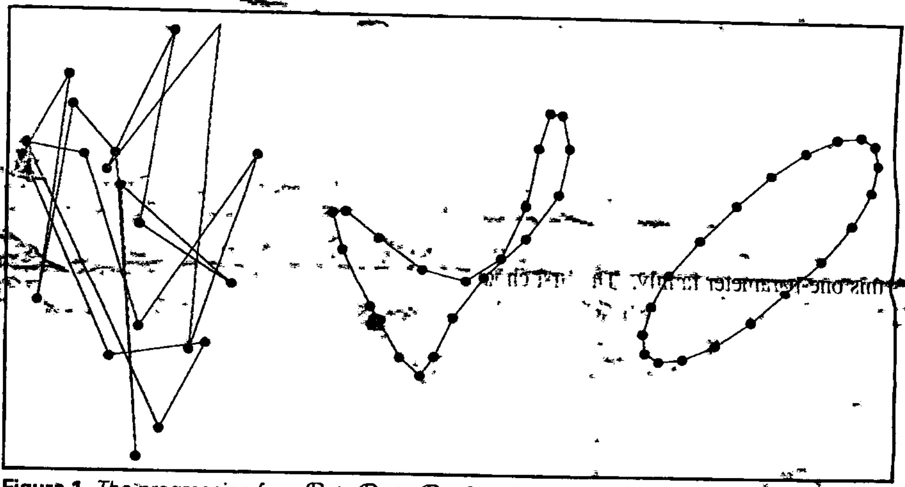

Figure 1. *The progression from $\mathcal{P}_0$ to $\mathcal{P}_{18}$ to $\mathcal{P}_{200}$ for an $n=20$ example. Figure courtesy of [1].*

My CS1 freshmen did not know enough linear algebra to think like this. But they did know enough MATLAB to watch the polygons untangle and ask the ultimate research-driven question: why?

On to the next course, a CS2 treatment of introductory matrix computations. Those students did know enough linear algebra to understand that the transition from one polygon to the next involves a pair of matrix-vector updates. In particular, the x and y vertex updates involve matrix-vector products $x\leftarrow Mx$ and $y\leftarrow My$, where $M=(I+S)/2$ and S is the upshift matrix, e.g.,

$$S=\begin{pmatrix}0&1&0&0\\0&0&1&0\\0&0&0&1\\1&0&0&0\end{pmatrix}.$$

With this notation, one can describe the polygon sequence's production as two separate instances of the power method. One produces the vector sequence $\{M^k x_{orig}\}$ and the other produces $\{M^k y_{orig}\}$, where x_{orig} and y_{orig} house the vertices of the original polygon $\mathcal{P}_0$. Because we know closed formulas for the eigenvalues and eigenvectors of the downshift matrix S (and hence M), we can develop closed-form expressions that completely specify the vertices for each and every polygon $\mathcal{P}_k$. Hint: they involve lots of sines and cosines. The CS2 polygon averaging assignment revolved around building insight for power method convergence associated with the matrix M.

The task then transitioned to a research project with Adam Elmachtoub, an undergraduate at Cornell University at the time. We determined how and why the polygon vertices moved towards an ellipse with 45-degree tilt. It turns out that the invariant subspace associated with the second- and third-largest eigenvalue of $M=(I+S)/2$ is critical to the analysis. The semiaxes and 45-degree tilt of the target ellipse follow from a singular value decomposition (SVD) analysis of a 2-by-2 matrix whose entries are simple functions of x_{orig} and y_{orig} (the singular values and left singular vectors of this 2-by-2 totally specify the tilted ellipse). Full details are available in the resulting *SIAM Review* paper [1].

The path from CS1 to *SIAM Review* was interesting from start to finish. We observed a phenomenon through experimentation and followed it up with a matrix-vector description of that phenomenon and an SVD/eigenvalue analysis that explained everything. Things do not always work out this nicely. Nevertheless, it is fun to think about the polygon averaging problem as simply a metaphor that speaks to the power of matrix-based scientific computing.

I will talk about this trajectory of our work at the John von Neumann Lecture at the 2018 SIAM Annual Meeting.

References

[1] Elmachtoub, A.N., & Van Loan, C.F. (2010). From Random Polygon to Ellipse: An Eigenanalysis. *SIAM Rev., 52*, 151-170.

Charles F. Van Loan is the 2018 recipient of SIAM's flagship John von Neumann Lecture. He is professor emeritus in the Department of Computer Science at Cornell University.