Editor: Dianne P. O'Leary, oleary@cs.umd.edu

ACHIEVING A COMMON VIEWPOINT: YAW, PITCH, AND ROLL

By Dianne P. O'Leary and David A. Schug

LIFE IS ABOUT CHANGE; NOTHING EVER STAYS THE SAME. IN PARTICULAR, OBJECTS MOVE, AND TRACKING THEM IS AN ESSENTIAL INGREDIENT IN APPLICATIONS SUCH AS NAVIGATION and robot motion. Surprisingly, the same mathematical tools used in tracking are also used in the *absolute orientation* problem of comparing two objects (such as proteins or machine parts) to see if they have the same structure. In this Homework Assignment, we develop the mathematical and computational tools to solve such problems.

Consider the molecule $\boldsymbol{A}$ in Figure 1, which we will specify by the coordinates $\mathbf{a}_1, \ldots, \mathbf{a}_7$ of the centers of the seven spheres that represent some of its atoms, and the corresponding object $\boldsymbol{B}$, obtained by rotating $\boldsymbol{A}$. There are many ways to define 3D rotations, but in this assignment, we'll specify *yaw* ϕ, *pitch* θ, and *roll* ψ, as is common in flight control. In this coordinate system, the angles ϕ, θ, and ψ are called the *Euler angles*, and a rotation $\boldsymbol{Q}$ is defined by the product of three matrices

$$\boldsymbol{Q}(\phi, \theta, \psi) = \boldsymbol{Q}_{roll}\, \boldsymbol{Q}_{pitch}\, \boldsymbol{Q}_{yaw},$$

where

$$Q_{roll} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & \sin\psi \\ 0 & -\sin\psi & \cos\psi \end{bmatrix},$$

$$Q_{pitch} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix},$$

$$Q_{yaw} = \begin{bmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

We impose the restrictions $-\pi < \phi \leq \pi$, $-\pi/2 < \theta \leq \pi/2$, and $-\pi < \psi \leq \pi$. Our first task is to develop some familiarity with this representation for rotation matrices.

PROBLEM 1.

a. Explain geometrically the effect of applying a rotation $\boldsymbol{Q}$ to a vector $[x, y, z]^T$ to create the vector $\boldsymbol{Q}\,[x, y, z]^T$.

b. Show that if $\boldsymbol{Q}$ is any 3×3 orthogonal matrix (i.e., $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}$, the identity matrix), then $\boldsymbol{Q}$ can be expressed as $\boldsymbol{Q}_{roll}\, \boldsymbol{Q}_{pitch}\, \boldsymbol{Q}_{yaw}$ for some choice of angles ψ, θ, and ϕ.

Next, we need to construct a way to determine the Euler angles when given data such as that in Figure 1. Let $\boldsymbol{A}$ be the $3 \times n$ matrix ($n = 7$) whose columns are the coordinates of the first set of points: $\boldsymbol{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_7]$. Define $\boldsymbol{B}$ similarly from the second set of points. Now we want to determine the three Euler angles so that

Tracking objects, controlling the navigation system of a spacecraft, assessing the quality of machined parts, and identifying proteins seem to have little in common, but all of these problems (and many more in computer vision and computational geometry) share a core computational task: rotating and translating two objects so that they have a common coordinate system. In this homework assignment, we study this deceptively simple computation and its pitfalls.

 Copublished by the IEEE CS and the AIP 1521-9615/04/$20.00 © 2004 IEEE

$B = Q(\phi, \theta, \psi)\, A$.

Because life is about change *and* imperfection, we don't expect to get an exact equality, but we want to make the difference between B and $Q(\phi, \theta, \psi)\, A$ as small as possible. One reasonable way to measure this is by taking the sum of the square of the differences in each component; then our task is to minimize

$$f(\phi, \theta, \psi) = \| B - Q(\phi, \theta, \psi)\, A \|_F^2 \equiv \sum_{i=1}^{n} \| \mathbf{b}_i - Q(\phi, \theta, \psi)\, \mathbf{a}_i \|_2^2,$$

where $\| . \|_F$ is called the *Frobenius norm* of the matrix. This is a nonlinear least-squares problem with three variables, so let's experiment with solving the problem for various data sets. The square root of f / n is the *root-mean-squared distance* (RMSD) between the two objects. The factor $1/n$ applied to f forms the average (mean) of the squared distances between the corresponding points. RMSD will provide us in Problem 2 with a measure of how well our objects match.

Problem 2.

Use a nonlinear least-squares solver to find the Euler angles for the data sets generated by taking the yaw $\phi = \pi/4$, roll $\psi = \pi/9$, and

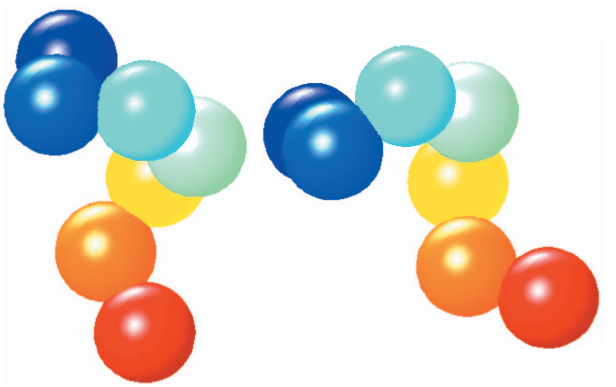
$$A = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & -1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 2 \\ 0 & 1 & 2 & 3 & 4 & 4 & 4 \end{bmatrix}.$$

Let the pitch θ vary between $-\pi/2$ and $\pi/2$ in steps of $\pi/120$. Plot the computed Euler angles, and, in a separate plot, the Frobenius norm of the error in Q and in the RMSD in the computed positions. Discuss the time needed for solution and the accuracy obtained.

The problem we are considering is an old one, sometimes called the *orthogonal Procrustes problem*. In Problem 3, we derive a better way to solve it.

Problem 3.

a. Recall that the trace *tr* of a square matrix is the sum of its main diagonal entries. We need two facts about traces in order to derive our algorithm. Prove that for any matrix C, $tr(C^T C) = \| C \|_F^2$, and that for any matrix D for which the product CD is defined, $tr(CD) = tr(DC)$.



(a) (b)

Figure 1. 3D rotations. How can we tell whether (a) molecule *A* and (b) molecule *B* are the same?

b. Use the first fact to show that the Q that minimizes $\| B - QA \|_F^2$ over all choices of orthogonal Q also maximizes $tr(A^T Q^T B)$.

c. Suppose that the singular value decomposition (SVD) of the $m \times m$ matrix BA^T is $U\Sigma V^T$, where U and V are $m \times m$ and orthogonal and Σ is diagonal with diagonal entries $\sigma_1 \geq \ldots \geq \sigma_m \geq 0$. Define $Z = V^T Q^T U$. Use these definitions and the second fact to show that

$$tr(A^T Q^T B) = tr(Q^T B A^T) = tr(Z\Sigma) \leq \sum_{i=1}^{m} \sigma_i \ .$$

d. If $Z = I$, then

$$tr(Q^T B A^T) = \sum_{i=1}^{m} \sigma_i \ .$$

What choice of Q ensures this?

Problem 3 shows that Q can be computed just by doing an SVD of BA^T, which is much more efficient than solving the nonlinear least-squares problem of Problem 2. Let's redo the computations.

Problem 4.

Use the SVD to find the Euler angles for the data in Problem 2. Compare with your previous results.

So far, we've assumed that the object has rotated with respect to the origin, but has not translated. Now we consider a more general problem:

$$B = Q(\phi, \theta, \psi)\, A + \mathbf{t}\mathbf{e}^T,$$

where the 3×1 vector $\mathbf{t}$ defines the translation, and $\mathbf{e}$ is a col-

Tools

One important problem that we've ignored is that of getting a set of corresponding points from the two objects. This is treated, for example, in Emanuele Trucco and Alessandro Verri's text.[1]

To help with Problem 1, you'll find a nice demonstration of the parameters for yaw, pitch, and roll at "How Things Fly" (www.nasm.si.edu/exhibitions/gal109/NEWHTF/ROLL.HTM). Other rotation coordinate systems are described in *Euler Angles*.[2]

In Problem 2, you can use Matlab's `lsqnonlin`.

If you get stuck in Problem 3, Gene Golub and Charles van Loan[3] give a helpful discussion of the orthogonal Procrustes.

Jack B. Kuipers[4] discusses the use of quaternions instead of Euler angles.

References

1. E. Trucco and A. Verri, *Introductory Techniques for 3D Computer Vision*, Prentice Hall, 1998.
2. E.W. Weisstein, *Euler Angles*, MathWorld, http://mathworld.wolfram.com/EulerAngles.html
3. G.H. Golub and C.F. Van Loan, *Matrix Computations*, 2nd ed., Johns Hopkins Press, 1989.
4. J.B. Kuipers, *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace and Virtual Reality*, Princeton Univ. Press, 2002.

umn vector with n ones. How might we solve this problem?

One way is to solve a nonlinear least-squares problem for $\mathbf{t}$ and the Euler angles. As in the May/June 2004 installment, we should take advantage of the fact that given $\mathbf{t}$, computing the optimal $\boldsymbol{Q}$ is easy, so we should express the problem as a function of just three variables: t_1, t_2, and t_3. Implementing this algorithm is interesting, but we'll just focus on a much more efficient approach.

The "easy" way arises from observing that the translation can be defined by the movement of the centroid of the points:

$$\mathbf{c}_A = \frac{1}{n}\sum_{j=1}^{n}\mathbf{a}_j ,$$

$$\mathbf{c}_B = \frac{1}{n}\sum_{j=1}^{n}\mathbf{b}_j .$$

Luckily, the averaging in the centroid computations tends to reduce the effects of random errors, and Problem 5 shows how $\mathbf{t}$ can be defined in terms of the centroids.

Problem 5.

Given a fixed rotation matrix $\boldsymbol{Q}$, show that the minimizer of $\|\boldsymbol{B} - \boldsymbol{QA} - \mathbf{te}^T\|_F$ satisfies $\mathbf{t} = \mathbf{c}_A - \boldsymbol{Q}\mathbf{c}_B$.

So we can solve our problem by moving both objects so that their centroids are at zero and then computing the resulting rotation $\boldsymbol{Q}$ using the SVD. Finally, we reconstruct the translation using the formula in Problem 5. Let's see how this algorithm behaves.

Problem 6.

Implement this algorithm and try it on the data from Problem 2 using $\theta = \pi/4$ and 20 randomly generated translations $\mathbf{t}$. Then repeat the experiment with 20 more translations, adding perturbations to the $\boldsymbol{A}$ data that are uniformly distributed between -10^{-3} and 10^{-3}, to see how sensitive the computation is to uncertainty in the measurements.

Through these computations (and further experimentation, if desired), you can see that the rotation matrix $\boldsymbol{Q}$ can almost always be computed quite accurately by the SVD algorithm; unfortunately, the Euler angles are not as well determined. In the next problem, we'll study these degenerate cases.

Problem 7.

a. Suppose that all of our points in $\boldsymbol{A}$ lie on a line. Is there more than one choice of $\boldsymbol{Q}$ that minimizes $\|\boldsymbol{B} - \boldsymbol{QA}\|$? Illustrate this with a numerical example.

b. Use this insight to characterize the degenerate cases for which $\boldsymbol{Q}$ is not well determined.

c. Suppose that our data produces the angles $(\phi, \theta = \pi/2, \psi)$, but a small perturbation causes a small increase in the angle θ so that it is greater than $\pi/2$. Generate such an example: you'll see that the computed angles are quite different. This jump in angle is called *gimbal lock*, a term borrowed from the locking of the mechanism that moves a stabilizing gyroscope in cases in which the angle goes out of the device's range of motion.

Thus, we can always choose a set of reference points to make the matrix $\boldsymbol{Q}$ well determined, but, unfortunately, this does not guarantee that the Euler angles are well determined.

One way to avoid this artificial ill-conditioning is to replace Euler angles with a better representation of the information in $\boldsymbol{Q}$. Quaternions are a common choice, and the "Tools" sidebar gives a pointer to more information on this subject.

Partial Solution to Last Issue's Homework Assignment

ELASTOPLASTIC TORSION: TWIST AND STRESS

By Dianne P. O'Leary

THE STANDARD MODEL INVOLVES THE *STRESS FUNCTION* $u(x, y)$, WHERE THE QUANTITIES $-\partial u(x, y)/\partial x$ AND $\partial u(x, y)/\partial y$ ARE THE STRESS COMPONENTS:

$$\nabla^2 u \equiv \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \equiv u_{xx} + u_{yy} = -2G\theta \text{ in } D,$$

$$u = 0 \text{ on } \hat{D},$$

where $\hat{D}$ is the boundary of the domain D, G is the material's *shear modulus*, and θ is the angle of twist per unit length.

We derive an alternate equivalent formulation by minimizing an energy function

$$E(u) = \frac{1}{2}\iint_D |\nabla u(x, y)|\, dx\, dy - 2G\theta \iint_D u(x, y)\, dx\, dy.$$

PROBLEM 1.

Suppose that the rod's cross-section D is the interior of a circle of radius one, and let $G = 5$ and $\theta = 1$. Use a finite-element package to approximate the stress function. Plot the approximate solution and describe what it says about the stress. Solve again using a finer mesh and estimate the error in your approximation to $1/2\mathbf{u}^T\mathbf{K}\mathbf{u} - \mathbf{b}^T\mathbf{u}$ to $E(u)$.

Answer: Sample Matlab code is available on the Web site (www.computer.org/cise/homework/). We can estimate the error in $E(u)$ by computing estimates with finer and finer grids, using the finest one as an approximation to truth. We expect the error in the estimates to drop by a factor of 4 each time the mesh size is halved (because the error is proportional to h^2), and that is what we observe. The mesh in Figure A produces an energy estimate with estimated error less than 0.1; Figure B shows the resulting solution.

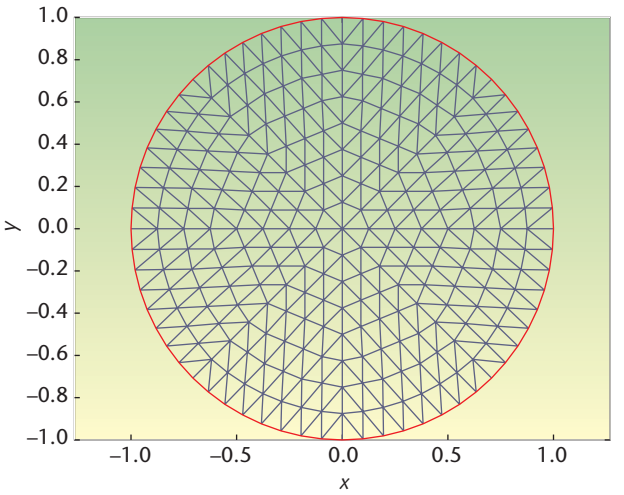


Figure A. Solution to Problem 1: the mesh used for a circular cross-section.

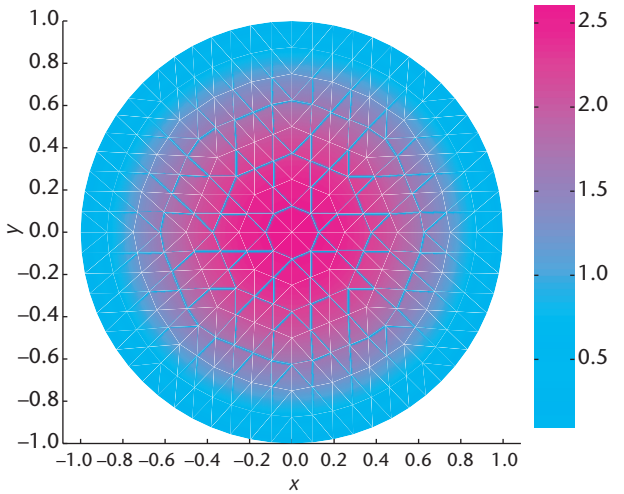


Figure B. Solution to Problem 1: the solution for the elastic model using a circular cross-section.

PROBLEM 2.

Derive an algorithm for finding the distance $d(\mathbf{z})$ between a given point $\mathbf{z} = [z_1, z_2]^T$ and an ellipse. In other words, solve the problem

$$\min_{x,y} (x - z_1)^2 + (y - z_2)^2,$$

Copublished by the IEEE CS and the AIP 1521-9615/04/$20.00 © 2004 IEEE

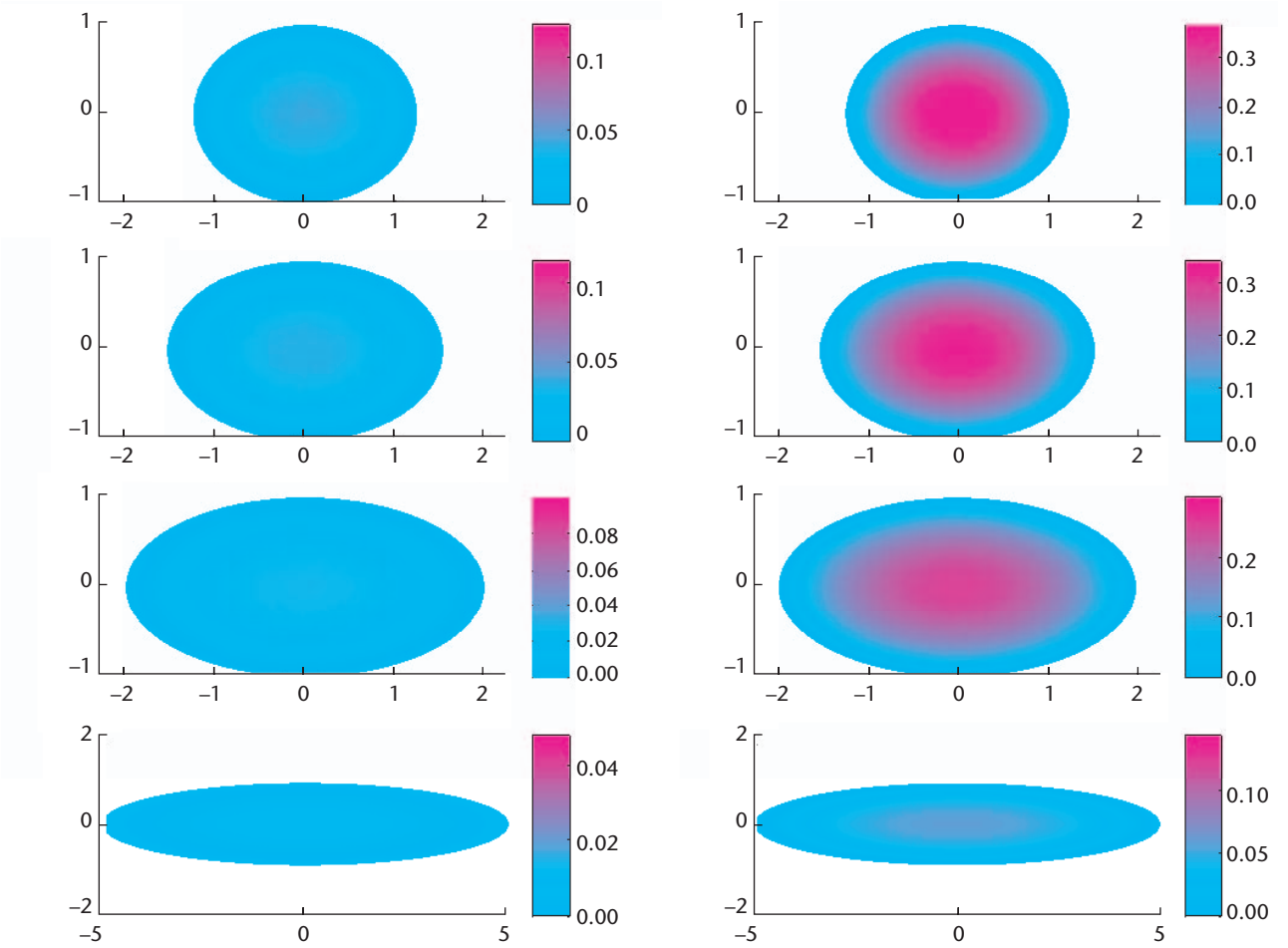


Figure C. Solution to Problem 4: elastoplastic solutions for various cross-sections. On the left, $\alpha\theta = 0.5$; on the right, $\alpha\theta = 1.0$.

subject to

$$\left(\frac{x}{\alpha}\right)^2 + \left(\frac{y}{\beta}\right)^2 = 1,$$

for given parameters α and β. Note that the distance is the square root of the optimal value of the objective function $(x - z_1)^2 + (y - z_2)^2$. The problem can be solved using Lagrange multipliers, as a calculus student would. You need only consider points **z** on or inside the ellipse, but handle all the special cases: $\alpha = \beta$, **z** has a zero coordinate, and so forth.

Answer: We set up the Lagrangian function

$$L(x, y, \lambda) = (x - z_1)^2 + (y - z_2)^2 - \lambda\left(\left(\frac{x}{\alpha}\right)^2 + \left(\frac{y}{\beta}\right)^2 - 1\right),$$

where the scalar λ is the Lagrange multiplier for the constraint. Setting the three partial derivatives to zero yields

$$2(x - z_1) - 2\lambda\frac{x}{\alpha^2} = 0,$$

$$2(y - z_2) - 2\lambda\frac{y}{\beta^2} = 0,$$

$$\left(\frac{x}{\alpha}\right)^2 + \left(\frac{y}{\beta}\right)^2 - 1 = 0.$$

We conclude that

$$x = \frac{\alpha^2 z_1}{\alpha^2 - \lambda}, \quad (1)$$

$$y = \frac{\beta^2 z_2}{\beta^2 - \lambda}, \quad (2)$$

as long as the denominators are nonzero. Because $|x| \le \alpha$ and $|y| \le \beta$, we conclude that the solution we seek has λ satisfying $0 \le \lambda \le \min(\alpha^2, \beta^2)$. So, we can solve our problem by solving the nonlinear equation

$$f(\lambda) = \left(\frac{x}{\alpha}\right)^2 + \left(\frac{y}{\beta}\right)^2 - 1 = 0,$$

using Equations 1 and 2 to define $x(\lambda)$ and $y(\lambda)$.

These formulas fail when $z_1 = 0$ or $z_2 = 0$. There are two points to check, depending on whether it is shorter to move horizontally or vertically to the boundary. When $\mathbf{z} = 0$, for example, then the solution is either $(x, y) = (0, \beta)$ or $(\alpha, 0)$, depending on whether β or α is smaller. Full details are given in the sample code for Problem 3 and in David Eberly's description.[1]

Problem 3.

Program your distance algorithm, document it, and produce a convincing validation of the code by designing a suitable set of tests and discussing the results.

Answer: Sample code appears on the Web site as `dist_to_ellipse.m`. The testing code plots the distances on a grid of points in the ellipse. Note that it's important to test the points that are near zero. To validate the code, we might repeat the runs with various values of α and β, and also test the code for a point $\mathbf{z}$ outside the ellipse.

Problem 4.

The elastoplastic model is

$$\min_u E(u)$$

$$|\nabla u(x, y)| \le \sigma_0,$$

$$u = 0 \text{ on } \hat{D}.$$

Solve the elastoplastic problem on a mesh that you estimate will give an error of less than 0.1 in the function $E(u)$. Use the parameters $G = 1$, $\sigma_0 = 1$, and $\beta = 1$. Let $\alpha\theta = 0$, 0.25, 0.50, …, 5 and $\beta/\alpha = 1$, 0.8, 0.65, 0.5, 0.2. Plot a few representative solutions. On a separate graph, for each value of β/α, plot a curve $T/(\sigma_0\alpha^3)$ versus $G\alpha\theta/\sigma_0$, where T is the estimate of the torque, the integral of u over the domain D.

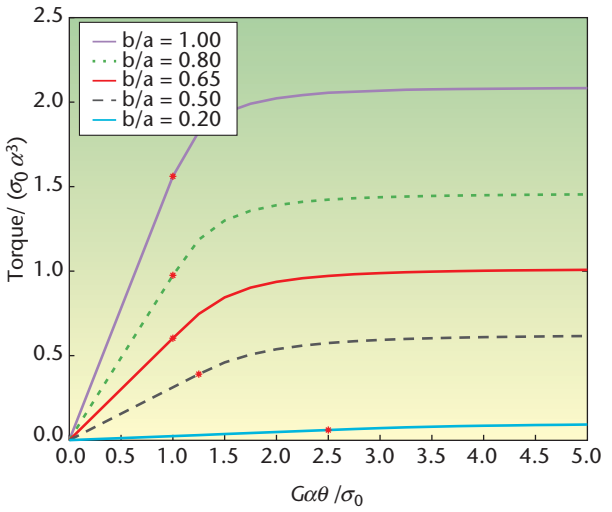


Figure D. Solution to Problem 4: torque computed for various cross-sections as θ is increased. The red stars mark the boundary between elastic solutions and elastoplastic solutions.

(This will give you five curves.) On the same plot, separate the elastic solutions (those for which no variable is at its bound) from the elastoplastic ones. Estimate the errors in your plot's data points.

Answer: Figures C and D show the results, which are computed with code on the Web site. The meshes we used had the same refinement as that determined for the circular domain in Problem 1. A sensitivity analysis should be done obtaining an error estimate by refining the mesh once to see how much the solution changes.

It would be more computationally efficient to take advantage of the sequence of problems being solved by using the solution at the previous value of $\alpha\theta$ as an initial guess for the next value.

Reference

1. D. Eberly, "Distance from a Point to an Ellipse in 2D," *Magic Software*, 2004, www.magic-software.com/Documentation/DistancePointToEllipse2.pdf.

Dianne P. O'Leary is a professor of computer science and a faculty member in the Institute for Advanced Computer Studies and the Applied Mathematics Program at the University of Maryland. She received a BS in mathematics from Purdue University and a PhD in computer science from Stanford. She is a member of SIAM, ACM, and AWM. Contact her at oleary@cs.umd.edu; www.cs.umd.edu/users/oleary/.

David A. Schug is an advanced special student at the University of Maryland, where he received a BS in mathematics. Contact him at david.schug@navy.mil.