Class Challenge • Models of Nonlinearity (math 990) • Lia Bronsard's Talk

- most work to be done during regular class periods (26,28 February).
- indicate work done beyond these hours (who & when).
- Professor Kropinski will be acting as consultant while I'm away.
- I will also be able to answer e-mails on an intermittent basis.

A) One of the equations that Lia Bronsard spoke about was a (complex-valued) phase-field model for a three-phase material in a two-dimensional domain $\mathcal{D}$. The equation for $z(x, y, t)$ is given in terms of a gradient flow

$$z_t = -\frac{\delta J}{\delta z^*} \qquad ; \qquad J[z, z*] = \int_{\mathcal{D}} F(z, z^* \ldots)\; dx\, dy \tag{1}$$

$$F(z, z_x, z_y, z^*, z_x^*, z_y^*) = \epsilon\, |\vec{\nabla} z|^2 + \frac{1}{\epsilon}\, |(z-A)(z-B)(z-C)|^2$$

where $J[z, z^*]$ is a real-valued functional of $z(x, y, t)$ and its complex conjugate $z^*(x, y, t)$. Note that the functional should be calculated as if z and z^* were independent, that is

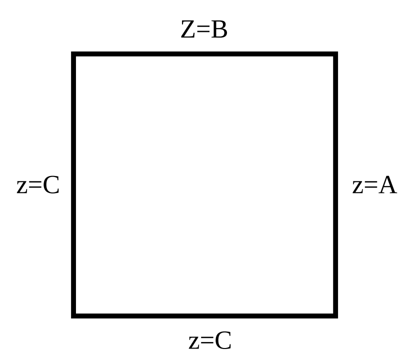
$$\frac{\delta J}{\delta z^*} = \left.\frac{\partial F}{\partial z^*}\right|_z - \frac{\partial}{\partial x}\left.\frac{\partial F}{\partial z_x^*}\right|_z - \frac{\partial}{\partial y}\left.\frac{\partial F}{\partial z_y^*}\right|_z \tag{2}$$

Absolute value bars indicate the magnitude of complex quantities. The complex-valued nature of the solution arises when the constants A, B, C are taken to be complex numbers; in Lia's example, these were the complex cube roots of unity.

Taking the variational gives an evolution of the form

$$z_t = \epsilon\, \nabla^2 z - \frac{1}{\epsilon}\left\{\, |(z-B)(z-C)|^2\, (z-A) + \ldots \right\} \tag{3}$$

Your mission ... is to write a matlab script that solves for the steady-states $\bar{z}(x, y)$ of this problem on a unit-square domain, when the boundary assumes only the values $z = A, B, C$ along piecewise sections of the perimeter. For example:



Although the regime of most interest for Professor Bronsard is the limit of $\epsilon \to 0$, you will see that this is also where the numerical problem becomes more difficult.

Methodology ... consider an iteration that solves for a sequence of $\bar{z}^n(x, y)$ where

$$\nabla^2 \bar{z}^{n+1} = \frac{1}{\epsilon^2} \text{ nonlinearity}(\bar{z}^n) \tag{4}$$

and the Laplacian is approximated by the second-order difference approximation. This essentially defines an iterative Poisson solve whose unknowns are the values of $\bar{z}^{n+1}$. The key piece of numerical software that is required is an efficient Poisson solver.

The place I began is a matlab demo called *delsqdemo* (it is a standard demo, you too should already have it). On the webpage is a modified version that I call *my_delsqdemo.m* which only runs the case that is closest to what you need to know – it is a Poisson solver, but only allows for zero boundary conditions. The basic Poisson inversion solves a set of linear equations

$$\frac{\bar{z}_{j-1,k} - 2\bar{z}_{j,k} + \bar{z}_{j+1,k}}{h^2} + \frac{\bar{z}_{j,k-1} - 2\bar{z}_{j,k} + \bar{z}_{j,k+1}}{h^2} = f_{j,k} \tag{5}$$

over all j, k in the interior of the unit square. Some data accounting is necessary to make the left-side of these equations into a matrix that will multiply a vector that is built from the 2D array of unknowns $\bar{z}_{j,k}$ (this is done using *numgrid*). Note that the html documentation isn't as complete as the matlab inline *help numgrid*.

I have written a Laplace solver which does allow for non-zero boundary conditions. The way it works is quite simple. If the above equation (5) involves a boundary point, I just move that term to the right-side (effectively modifying the $f_{j,k}$) and then use a Poisson solver for zero boundaries! The code that does this is *pois.m* – both this and *my_delsqdemo.m* are available from the class webpage.

The purpose of this exercise ... is to involve the *whole* class in this numerical development. By the end of the week, everyone should be comfortable with any resulting codes. Once you have figured out how *pois.m* works, I expect that as a group you can design an iteration routine to solve Lia's problem for $\epsilon = 1$. This approach seems to get bogged down once $\epsilon \approx 0.38$.

Good luck!