Gerda de Vries, Arthur Sherman, and Hsiu-Rong Zhu, *Diffusively Coupled Bursters: Effects of Cell Heterogeneity*, Bulletin of Mathematical Biology, 60 (1998), pp. 1167–1200.

Appendix

Equation numbers in the following refer to the equations as they appear in the published version of the paper.

Reduction to normal form

In this section, we present the Mathematica code which implements the algorithm for determining the values of α, β, and η from (29) via (20)–(24).

We apply the algorithm to the fast subsystem of the nondimensional version of (8)–(10). Our nondimensionalization is based on letting $v = 25x - 225$, $n = y$, $s = \lambda z/4$, and $t = \tau \bar{t}/\lambda$.

Note that the values of the model parameters λ and g_{Ca} are left unspecified in order to be able to explore the dependence of α, β, and η on these parameters. Accordingly, `xguess`, the initial guess for the value of x at the Hopf bifurcation, is left unspecified as well.

```
(* Define the oscillatory system, dx/dt = f[x,y;z], dy/dt = g[x,y;z] *)
minf[x_] := 1/(1+Exp[(205-25x)/12]);
ninf[x_] := 1/(1+Exp[(208-25x)/5.6]);
f[x_,y_,z_] := -gca minf[x] (x-10)/lambda - 10 y (x-6)/lambda - z (x-6);
g[x_,y_,z_] := ninf[x] - y;

(* Define the steady state for the above system *)
yss[x_] := ninf[x];
zss[x_] := ( -gca minf[x] (x-10)/lambda - 10 ninf[x] (x-6)/lambda ) / (x-6);

(* Define the coupling matrix *)
dcoup = {{1,0},{0,0}};

vars = {x,y};
jac = Outer[D,{f[x,y,z],g[x,y,z]},vars];
m = Outer[D,jac,vars];
n = Outer[D,m,vars];

(* Determine the location of the Hopf bifurcation *)
trjac = jac[[1,1]] + jac[[2,2]] /. {y->yss[x], z->zss[x]};
x0 = x /. FindRoot[ trjac==0, {x,xguess} ];
y0 = yss[x0];
zHB = zss[x0];
```

```
l0 = jac /. {x->x0,y->y0,z->zHB};
m0 = m/2 /. {x->x0,y->y0,z->zHB};
n0 = n/6 /. {x->x0,y->y0,z->zHB};
(* Use the Implicit Function Theorem to find l1 *)
rhs = { -D[f[x,y,z],z], -D[g[x,y,z],z] };
{dxdz,dydz} = LinearSolve[l0,rhs];
l1 = D[jac,x] dxdz + D[jac,y] dydz + D[jac,z] /. {x->x0,y->y0,z->zHB};

(* Determine the eigenvalue, the corresponding eigenvectors u
   and ustar (normalized), and the derived quantities vplus, vminus *)
esystem1 = Eigensystem[l0];
esystem2 = Eigensystem[-Transpose[l0]];
omega = Im[ esystem1[[1,1]] ];
u = esystem1[[2,1]];
ustar = esystem2[[2,2]];
ustar /= ustar.u;
vplus = -LinearSolve[(l0-2 omega I IdentityMatrix[2]), m0.u.u];
vknot = -2 LinearSolve[l0,m0.u.Conjugate[u]];

(* Determine a + i b, p + i q, and d1 + i d2 *)
ab = ustar.l1.u;
pq = -( 2 ustar.m0.u.vknot +
        2 ustar.m0.Conjugate[u].vplus +
        3 ustar.n0.u.u.Conjugate[u] );
d1d2 = ustar.dcoup.u;

alpha = Im[d1d2] / Re[d1d2];
beta = Im[ab] / Re[ab];
eta = Im[pq] / Re[pq];
```

Analysis of the pitchfork bifurcation

In this section, we present the Mathematica code which implements the algorithm for determining the value of γ_2 from (52). The determination of L_1, M_0, N_0, $\mathbf{\Phi}_0$ and $\mathbf{\Psi}_0$ is relatively straightforward. The determination of $\mathbf{u}_2$ satisfying (53) is slightly more difficult, since L_0 has a zero eigenvalue and therefore is singular. We use a variant of the Singular Value Decomposition method to determine $\mathbf{u}_2$. In particular, we write

$$L_0 = P.[\mathrm{diag}(\lambda_i)].P^{-1}, \tag{1}$$

where the columns of P are the normalized eigenvectors of L_0, and the λ_i are the eigenvalues of L_0. Letting $\mathbf{v}_i$, $i = 1, \ldots, n$, denote the columns of P and $\mathbf{w}_i$, $i = 1, \ldots, n$, denote the

rows of P^{-1}, the solution to (53) then can be written as

$$\mathbf{u}_2 = \sum_{i=1,\lambda_i \neq 0}^{n} \frac{\mathbf{w}_i.\mathbf{b}}{\lambda_i}\mathbf{v}_i, \tag{2}$$

where $\mathbf{b} = -M_0\mathbf{\Phi}_0\mathbf{\Phi}_0$. To determine the $\mathbf{w}_i$'s, we use the fact that

$$P^{-1} = R^T, \tag{3}$$

where the columns of R are the eigenvectors of the adjoint of L_0, normalized to the $\mathbf{v}_i$'s.

We note that the value of γ_2 depends on the values of the parameters α and η, which are left unspecified in the code below.

```
(* The reduced system at Delta=1 *)
h1[r1_,r2_,phi_,gamma_] := r1 (1-r1^2) +
                           gamma (Cos[phi] - alpha Sin[phi]) r2 -
                           gamma r1;
h2[r1_,r2_,phi_,gamma_] := r2 (1-r2^2) +
                           gamma (Cos[phi] + alpha Sin[phi]) r1 -
                           gamma r2;
h3[r1_,r2_,phi_,gamma_] := eta (r1^2-r2^2) -
                           gamma r2/r1 (Sin[phi] + alpha Cos[phi]) -
                           gamma r1/r2 (Sin[phi] - alpha Cos[phi]);

vars = {r1, r2, phi};
jac = Outer[D,{h1[r1,r2,phi,gamma],h2[r1,r2,phi,gamma],h3[r1,r2,phi,gamma]},vars ];
m = Outer[D,jac,vars];
n = Outer[D,m,vars];

If[ 1 + alpha eta < 0.0,
    (* the pitchfork is on the in-phase branch *)
    gammaPF = -(1.0+alpha eta)/(1.0+alpha^2);
    r10 = 1.0;
    r20 = 1.0;
    phi0 = 0.0,
    (* the pitchfork is on the anti-phase branch *)
    gammaPF = (1.0+alpha eta)/(3.0+alpha^2+2.0 alpha eta);
    r10 = Sqrt[1.0-2.0 gammaPF];
    r20 = Sqrt[1.0-2.0 gammaPF];
    phi0 = N[Pi,10] ];

l0 = jac /. {r1->r10, r2->r20, phi->phi0, gamma->gammaPF};
If[ 1 + alpha eta < 0.0,
```

```
    (* the pitchfork is on the in-phase branch, and none of the
       steady-state coordinates depend on gamma *)
    l1 = D[jac,gamma] /. {r1->r10, r2->r20, phi->phi0, gamma->gammaPF},
    (* the pitchfork is on the anti-phase branch, and the r1 and r2
       coordinates of the steady state depend on gamma *)
    l1 = -( D[jac,r1] + D[jac,r2] ) / Sqrt[1-2 gamma] +
         D[jac,gamma] /. {r1->r10, r2->r20, phi->phi0, gamma->gammaPF} ];
m0 = m/2 /. {r1->r10, r2->r20, phi->phi0, gamma->gammaPF};
n0 = n/6 /. {r1->r10, r2->r20, phi->phi0, gamma->gammaPF};

(* Determine the eigenvectors of l0, and normalize them.  *)
{vals,vecs} = Eigensystem[l0];
For[ i=1, i<=3 && Abs[vals[[i]]] > 0.0001, i++ ];
phi0 = vecs[[i]];
indices = Complement[{1,2,3},{i}];
v1 = vecs[[Part[indices,1]]];
v2 = vecs[[Part[indices,2]]];
phi0 /= Sqrt[phi0.phi0];
v1 /= Sqrt[v1.v1];
v2 /= Sqrt[v2.v2];
evalue1 = vals[[Part[indices,1]]];
evalue2 = vals[[Part[indices,2]]];

(* Determine the eigenvectors of the adjoint of l0, and
   normalize them with respect to the eigenvectors of l0.  *)
{vals,vecs} = Eigensystem[Transpose[l0]];
For[ i=1, i<=3 && Abs[vals[[i]]] > 0.0001, i++ ];
psi0 = vecs[[i]];
indices = Complement[{1,2,3},{i}];
If[ vals[[Part[indices,1]]] == evalue1,
    w1 = vecs[[Part[indices,1]]];
    w2 = vecs[[Part[indices,2]]],
    w1 = vecs[[Part[indices,2]]];
    w2 = vecs[[Part[indices,1]]] ];
psi0 /= psi0.phi0;
w1 /= w1.v1;
w2 /= w2.v2;

(* Determine u2, satisfying the equation l0.u2 = -m0.phi0.phi0.
   Since l0 has a zero eigenvalue, l0 is singular.  To determine
   u2, we use the Singular Value Decomposition method.  *)
b = -m0.phi0.phi0;
```

```
u2 = w1.b/evalue1 v1 + w2.b/evalue2 v2;

gamma2 = -(2 m0.phi0.u2 + n0.phi0.phi0.phi0).psi0 / (psi0.l1.phi0)
```