Contents


```
function [M_best E_best] = GOOPSI_main_v1_0(F,P,Sim)
% this function runs the SMC-EM on a fluorescence time-series, and outputs the inferred
% distributions and parameter estimates
%
% Inputs
% F:    fluorescence time series
% P:    structure of initial parameter estimates
% Sim:  structure of stuff necessary to run smc-em code
%
% Outputs
% M_best:   structure containing mean, variance, and percentiles of inferred distributions
% E_best:   structure containing the final parameter estimates

i           = 0;            % iteration number of EM
k           = 0;            % best iteration so far
P.lik       = -inf;         % we are trying to maximize the likelihood here
maxlik      = P.lik;        % max lik achieved so far
F           = max(F,eps);   % in case there are any zeros in the F time series
Sim.conv    = false;        % EM has NOT yet converged.

if Sim.Mstep==1 && (~isfield(Sim,'SuppressGraphics') || Sim.SuppressGraphics == 0)
    figure(1), clf, nrows=4;
end % if estimating parameters, plot stuff for each iteration

cnt=0;
while Sim.conv==false;

    % some nomenclature to make code easier to read/write
    % these abbrev's are used in forward_step and backward_step
    P.a         = Sim.dt/P.tau_c;
    P.sig2_c    = P.sigma_c^2*Sim.dt;
    P.kx        = P.k'*Sim.x;
    if Sim.M==1
        P.sig2_h    = P.sigma_h.^2*Sim.dt;
        P.g         = 1-Sim.dt/P.tau_h;
    end
```

forward step

```
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
fprintf('\nT = %g steps',Sim.T)
fprintf('\nforward step:       ')
S = GOOPSI_forward_v1_0(Sim,F,P);
```

backward step

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fprintf('\nbackward step:       ')
Z.oney  = ones(Sim.N,1);                    % initialize stuff for speed
Z.zeroy = zeros(Sim.N);
Z.C0    = S.C(:,Sim.T);
Z.C0mat = Z.C0(:,Z.oney)';

if Sim.C_params==false                      % if not maximizing the calcium parameters,
    for t=Sim.T-Sim.freq-1:-1:Sim.freq+1      % actually recurse backwards for each time
        Z = GOOPSI_backward_v1_0(Sim,S,P,Z,t);
        S.w_b(:,t-1) = Z.w_b;                   % update forward-backward weights
    end
else                                        % if maximizing calcium parameters,
    % need to compute some sufficient statistics
    M.Q = zeros(3);                           % the quadratic term for the calcium par
    M.L = zeros(3,1);                         % the linear term for the calcium par
    M.J = 0;                                  % remaining terms for calcium par
    M.K = 0;
    for t=Sim.T-Sim.freq-1:-1:Sim.freq+1
        Z = GOOPSI_backward_v1_0(Sim,S,P,Z,t);
        S.w_b(:,t-1) = Z.w_b;

        % below is code to quickly get sufficient statistics
        C0dt    = Z.C0*Sim.dt;
        bmat    = Z.C1mat-Z.C0mat';
        bPHH    = Z.PHH.*bmat;

        M.Q(1,1)= M.Q(1,1) + sum(Z.PHH*(C0dt.^2));  % Q-term in QP
        M.Q(1,2)= M.Q(1,2) - Z.n1'*Z.PHH*C0dt;
        M.Q(1,3)= M.Q(1,3) + sum(sum(-Z.PHH.*Z.C0mat'*Sim.dt^2));
        M.Q(2,2)= M.Q(2,2) + sum(Z.PHH'*(Z.n1.^2));
        M.Q(2,3)= M.Q(2,3) + sum(sum(Z.PHH(:).*repmat(Z.n1,Sim.N,1))*Sim.dt);
        M.Q(3,3)= M.Q(3,3) + sum(Z.PHH(:))*Sim.dt^2;

        M.L(1)  = M.L(1) + sum(bPHH*C0dt);          % L-term in QP
        M.L(2)  = M.L(2) - sum(bPHH'*Z.n1);
        M.L(3)  = M.L(3) - Sim.dt*sum(bPHH(:));

        M.J     = M.J + sum(Z.PHH(:));              % J-term in QP /sum J^(i,j)_{t,t-1}/
```

```
        M.K     = M.K + sum(Z.PHH(:).*bmat(:).^2);  % K-term in QP /sum J^(i,j)_{t,t-1}
    end
    M.Q(2,1) = M.Q(1,2);                            % symmetrize Q
    M.Q(3,1) = M.Q(1,3);
    M.Q(3,2) = M.Q(2,3);
end

fprintf('\n')

%%%%%%%%%%%%%% HACK %%%%%%%%%%%%%%%%%
if(isfield(Sim,'TrueSpk'))                  % force true spikes hack
    M.n_sampl=S.n;
    S.n=repmat(Sim.TrueSpk(:)',[size(S.n,1) 1]);
end;
%%%%%%%%%%%%%% HACK %%%%%%%%%%%%%%%%%

% copy particle swarm for later
M.w=S.w_b;
M.n=S.n;
M.C=S.C;

% check failure mode caused by too high P.A (low P.sigma_c)
fact=1.55;
if(sum(S.n(:))==0 && cnt<10)                % means no spikes anywhere
    fprintf(['Failed to find any spikes, likely too high a P.A.\n',...
        'Attempting to lower by factor %g...\n'],fact);
    P.A=P.A/fact;
    P.C_0=P.C_0/fact;
    P.sigma_c=P.sigma_c/fact;
    cnt=cnt+1;
    continue;
elseif(cnt>=10)
    error('There are no spikes in the data. Wrong initialization? [Fatal]');
end
```

compute moments and percentiles

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if Sim.SuppressGraphics == 0
    % means
    M.nbar = sum(S.w_b.*S.n,1);
    M.Cbar = sum(S.w_b.*S.C,1);
    M.pbar = sum(S.w_b.*S.p,1);

    % variances
    M.nvar = sum((repmat(M.nbar,Sim.N,1)-S.n).^2)/Sim.N;
```

```
    M.Cvar = sum((repmat(M.Cbar,Sim.N,1)-S.C).^2)/Sim.N;
    M.pvar = sum((repmat(M.pbar,Sim.N,1)-S.p).^2)/Sim.N;

    % percentiles
    ptiles    = 1/Sim.N : 1/Sim.N : 1-1/Sim.N;
    M.Cptiles = GetPercentiles(ptiles,S.w_b,S.C);
    M.pptiles = GetPercentiles(ptiles,S.w_b,S.p);

    if Sim.M==1
        M.hbar = sum(S.w_b.*S.h,1);
        M.hvar = sum((repmat(M.hbar,Sim.N,1)-S.h).^2)/Sim.N;
        M.hptiles = GetPercentiles(ptiles,S.w_b,S.h);
    end
end
```

M step

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if Sim.Mstep
    i    = i+1;                        % increase iteration of EM
    Eold = P;                          % store most recent parameter structure
    P    = GOOPSI_Mstep_v1_0(Sim,S,M,P,F);% update parameters
    fprintf('\n\nIteration #%g, lik=%g, dlik=%g\n',i,P.lik,P.lik-Eold.lik)

    % keep record of best stuff, or if told to ignore lik
    if((isfield(P,'ignorelik') && P.ignorelik==1) || P.lik>= maxlik)
        E_best  = P;                   % update best parameters
        M_best  = M;                   % update best moments
        maxlik  = P.lik;               % update best likelihood
        k       = i;                   % save iteration number of best one
        if(~isfield(Sim,'SuppressGraphics') || ~Sim.SuppressGraphics)
            subplot(nrows,1,4), cla,hold on,% plot spike train estimate
            if isfield(Sim,'n'), stem(Sim.n,'Marker','.',...
                    'MarkerSize',20,'LineWidth',2,'Color',[.75 .75 .75]); end
            BarVar=M.nbar+M.nvar; BarVar(BarVar>1)=1;
            stem(BarVar,'Marker','none','LineWidth',2,'Color',[.8 .8 0]);
            stem(M.nbar,'Marker','none','LineWidth',2,'Color',[0 .5 0])
            axis([0 Sim.T 0 1]),
        end
    end

    % when estimating calcium parameters, display param estimates and lik
    if Sim.C_params==1
        dtheta  = norm([P.tau_c; P.A; P.C_0]-...
            [Eold.tau_c; Eold.A; Eold.C_0])/norm([Eold.tau_c; Eold.A; Eold.C_0; P.sigma_
        fprintf('\ndtheta = %.2f',dtheta);
```

```
            fprintf('\ntau    = %.2f',P.tau_c)
            fprintf('\nA      = %.2f',P.A)
            fprintf('\nC_0    = %.2f',P.C_0)
            fprintf('\nsig    = %.2f',P.sigma_c)
            fprintf('\nalpha  = %.2f',P.alpha)
            fprintf('\nbeta   = %.2f',P.beta)
            fprintf('\ngamma  = %.2g',P.gamma)
        end

        % plot lik and inferrence
        if(~isfield(Sim,'SuppressGraphics') || ~Sim.SuppressGraphics)
            if Sim.n_params == true
                fprintf('\nk      = %.2f',P.k)
            end
            subplot(nrows,1,1), hold on, plot(i,P.lik,'o'), axis('tight')
            subplot(nrows,1,2), plot(F,'k'), hold on,
            plot(P.alpha*Hill_v1(P,M.Cbar)+P.beta,'b'), hold off, axis('tight')
            subplot(nrows,1,3), cla, hold on,   % plot spike train estimate
            axis([0 Sim.T 0 1]),
            if isfield(Sim,'n'),
                stem(Sim.n,'Marker','.','MarkerSize',20,'LineWidth',2,...
                    'Color',[.75 .75 .75],'MarkerFaceColor','k','MarkerEdgeColor','k');
                axis('tight'),
            end
            BarVar=M.nbar+M.nvar; BarVar(BarVar>1)=1;
            stem(BarVar,'Marker','none','LineWidth',2,'Color',[.8 .8 0]);
            stem(M.nbar,'Marker','none','LineWidth',2,'Color',[0 .5 0])

            drawnow
        end

        if i>=Sim.MaxIter
            Sim.conv=true;
        end

    else
        M_best  = M;                     % required for output of function
        E_best  = P;
        Sim.conv= true;
    end

    E_best  = P;                     % update best parameters
    M_best  = M;                     % update best moments

end
if Sim.SuppressGraphics==1, M_best.nbar = sum(S.w_b.*S.n,1); end
```

```
Input argument "F" is undefined.

Error in ==> GOOPSI_main_v1_0 at 18
F           = max(F,eps);   % in case there are any zeros in the F time series
```