```
function x = IRKTemplate(ButcherArray, f, dfdx, T, x0)
    % Returns the iterations of an IRK method using Newton's method
    % ButcherArray: Struct with the IRK's Butcher array
    % f: Function handle
    %    Vector field of ODE, i.e., x_dot = f(t,x)
    % dfdx: Function handle
    %       Jacobian of f w.r.t. x
    % T: Vector of time points, 1 x Nt
    % x0: Initial state, Nx x 1
    % x: IRK iterations, Nx x Nt
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Define variables
    % Allocate space for iterations (x) and k1,k2,...,ks
    %
    %
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    tol=10^(-3);
    A=ButcherArray.A;
    b=ButcherArray.b;
    c=ButcherArray.c;
    x=zeros(length(x0),length(T));
    x(:,1) = x0; % initial iteration
    k = zeros(length(x0),length(A)); % initial guess
    % Loop over time points
    %r=IRKODEResidual(k(1,:),xt(:,1),1,del_t,A,c,f);
    r=1000;
    N=1000;
    Nt=length(T);
    alpha=1;
    for nt=2:Nt
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        % Update variables
        % Get the residual function for this time step
        % and its Jacobian by defining adequate functions
        % handles based on the functions below.
        % Solve for k1,k2,...,ks using Newton's method
        % Calculate and save next iteration value x_t
        %
        %
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        del_t = T(nt) - T(nt - 1);
        K = x(:,nt)*ones(length(x), length(A));
        k_reshape=reshape(k,[length(x0)*length(A),1]);
        r=IRKODEResidual(k_reshape,x(:,nt-1),nt,del_t,A,c,f);
        while norm(r)>tol
            r=IRKODEResidual(k,x(:,nt-1),nt,del_t,A,c,f);
            J_r=IRKODEJacobianResidual(k,x(:,nt-1),nt,del_t,A,c,dfdx);
            d=@(del_k) J_r*del_k+r;
            J=@(del_k) J_r;
            del_k=NewtonsMethod(d,J,k',tol,N);
            k=k+alpha*del_k';
            k_reshape=reshape(k,[length(x0)*length(A),1]);
        end
        x(:,nt)=x(:,nt-1)+(del_t*b*k');
    end
end
function g = IRKODEResidual(k,xt,t,dt,A,c,f)
    % Returns the residual function for the IRK scheme iteration
    % k: Column vector with k1,...,ks, Nstage*Nx x 1
    % xt: Current iteration, Nx x 1
    % t: Current time
    % dt: Time step to next iteration
    % A: A matrix of Butcher table, Nstage x Nstage
    % c: c matrix of Butcher table, Nstage x 1
    % f: Function handle for ODE vector field
    Nx = length(xt);
    Nstage = size(A,1);
    K = reshape(k,Nx,Nstage);
    Tg = t+dt*c';
    Xg = xt+dt*K*A';
    g = reshape(K-f(Tg,Xg),[],1);
end
function G = IRKODEJacobianResidual(k,xt,t,dt,A,c,dfdx)
    % Returns the Jacobian of the residual function
    % for the IRK scheme iteration
    % k: Column vector with k1,...,ks, Nstage*Nx x 1
    % xt: Current iteration, Nx x 1
    % t: Current time
    % dt: Time step to next iteration
    % A: A matrix of Butcher table, Nstage x Nstage
    % c: c matrix of Butcher table, Nstage x 1
    % dfdx: Function handle for Jacobian of ODE vector field
    Nx = length(xt);
    Nstage = size(A,1);
    K = reshape(k,Nx,Nstage);
    TG = t+dt*c';
    XG = xt+dt*K*A';
    dfdxG = cell2mat(arrayfun(@(i) dfdx(TG(:,i),XG(:,i))',1:Nstage,...
        'UniformOutput',false))';
    G = eye(Nx*Nstage)-repmat(dfdxG,1,Nstage).*kron(dt*A,ones(Nx));
end
```

```
Not enough input arguments.

Error in IRKTemplate (line 18)
    A=ButcherArray.A;
```

Published with MATLAB® R2019a