**Parareal implementation**

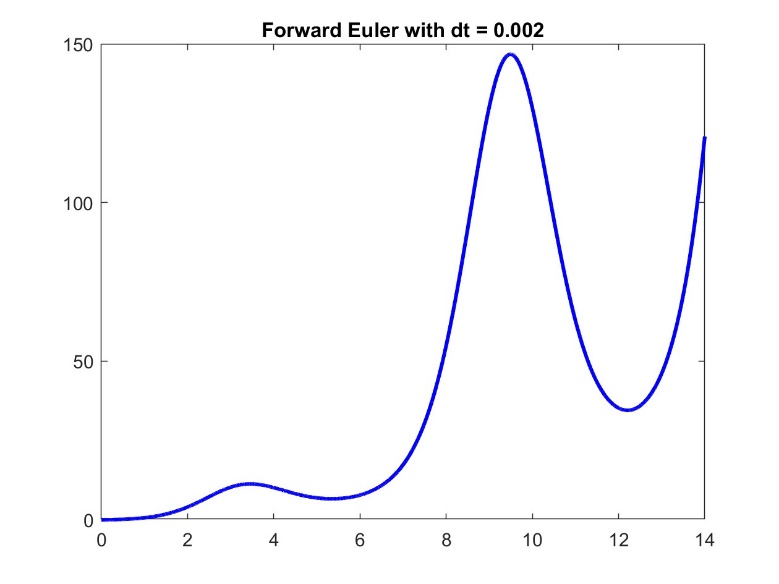
Consider the following ODE:

Introduce the following propagation operator:

* **Coarse propagator** (e.g. Forward Euler with )
* **Fine propagator** (e.g. Forward Euler with )

Notation: at the step , with

**My parareal**



**Step ~ 0**

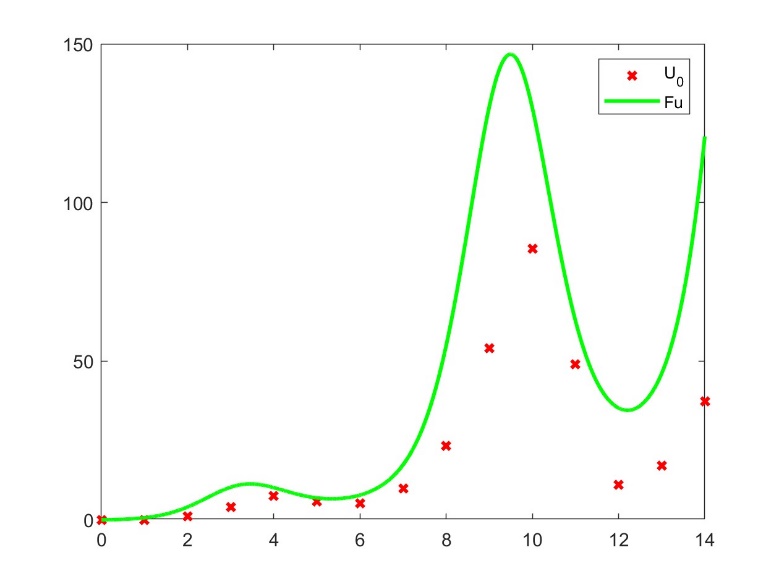
f = @(t,y) sin(t).\*y + t;

t0 = 0; tN = 14;

y0 = 0;

dt = 0.002;

[t,y\_fine] = fwd\_Euler(t0,tN,y0,dt,f);

****

**Step 1**

Initial guess, first iteration of

dT = 1;

[t\_coarse,U\_0] = fwd\_Euler(t0,tN,u0,dT,f);

**Definitions of u\_fine and t\_fine**

for m = 1 : (tN - t0) / dT % from 1 to number of coarse subintervals

t\_fine(m,:) = [t0 + (m-1) \* dT : dt : t0 + m \* dT]; % fine approx. of each subint.

end

% Notice: t\_fine(m,end) = t\_fine(m+1,1)

% L\_coarse = n of coarse subintervals, L\_fine = n of fine subinterval

u\_fine = zeros(L\_coarse, L\_fine);

U = U\_0;

**Iterative Step**

for k = 1 : k\_max

Between two successive coarse time steps

% Evaluation of

% Parallel step: computation of the fine solution on each subinterval

parfor n = 1 : L\_coarse - 1

[t\_fine(n,:),u\_fine(n,:)] = fwd\_Euler(t\_coarse(n),t\_coarse(n+1),U\_0(n),dt,f);

end

% Evaluation of

% Update of the coarse solution

for h = 1 : L\_coarse - 1

du = sin(t\_coarse(h))\*U(h) + t\_coarse(h);

U\_k(h+1) = U(h) + dT\*du;

end

% Evaluation of

for n = 1 : L\_coarse - 1

du = sin(t\_coarse(n))\*U(n) + t\_coarse(n);

U(n+1) = U(n) + dT\*du;

% Correction step

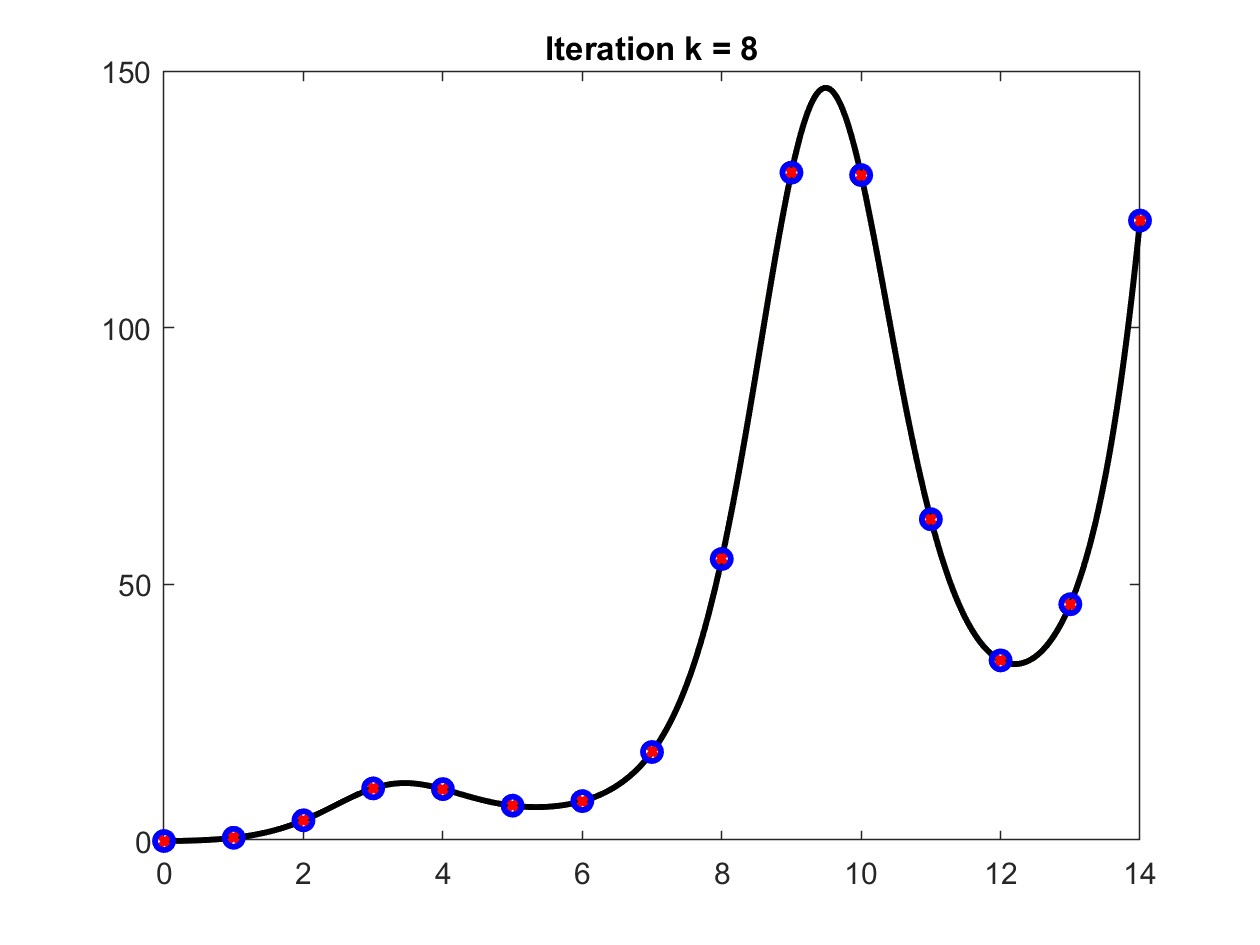
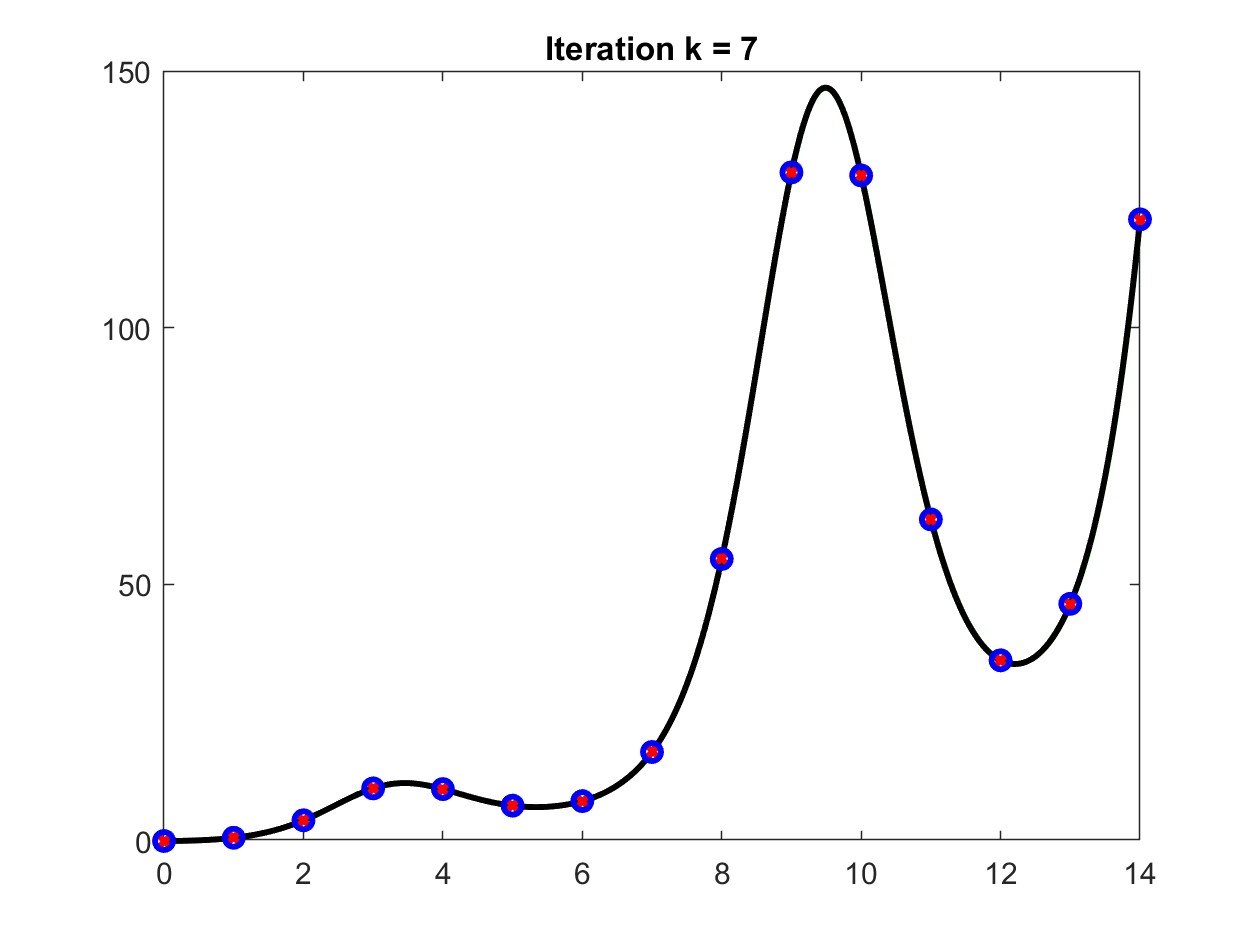
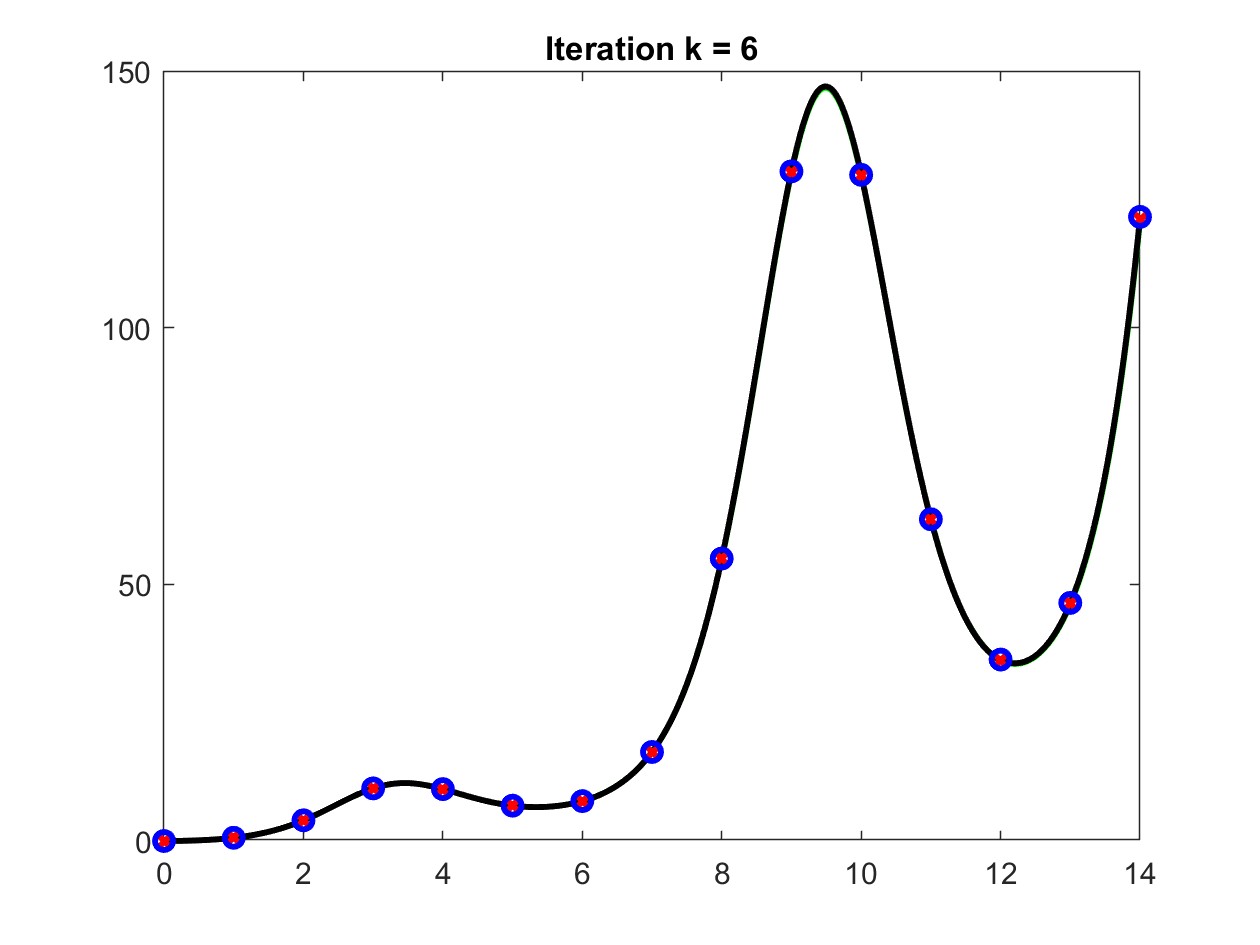
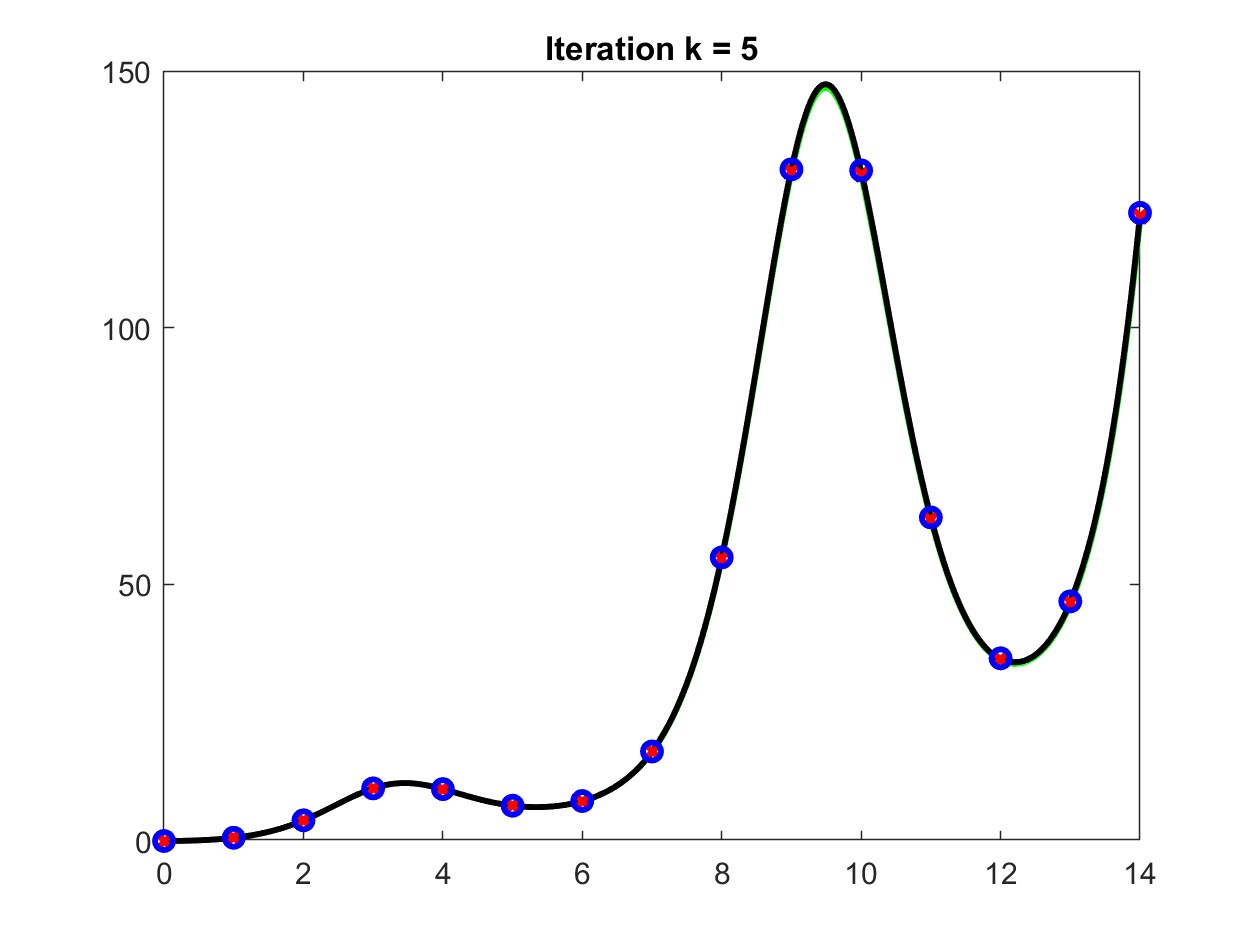
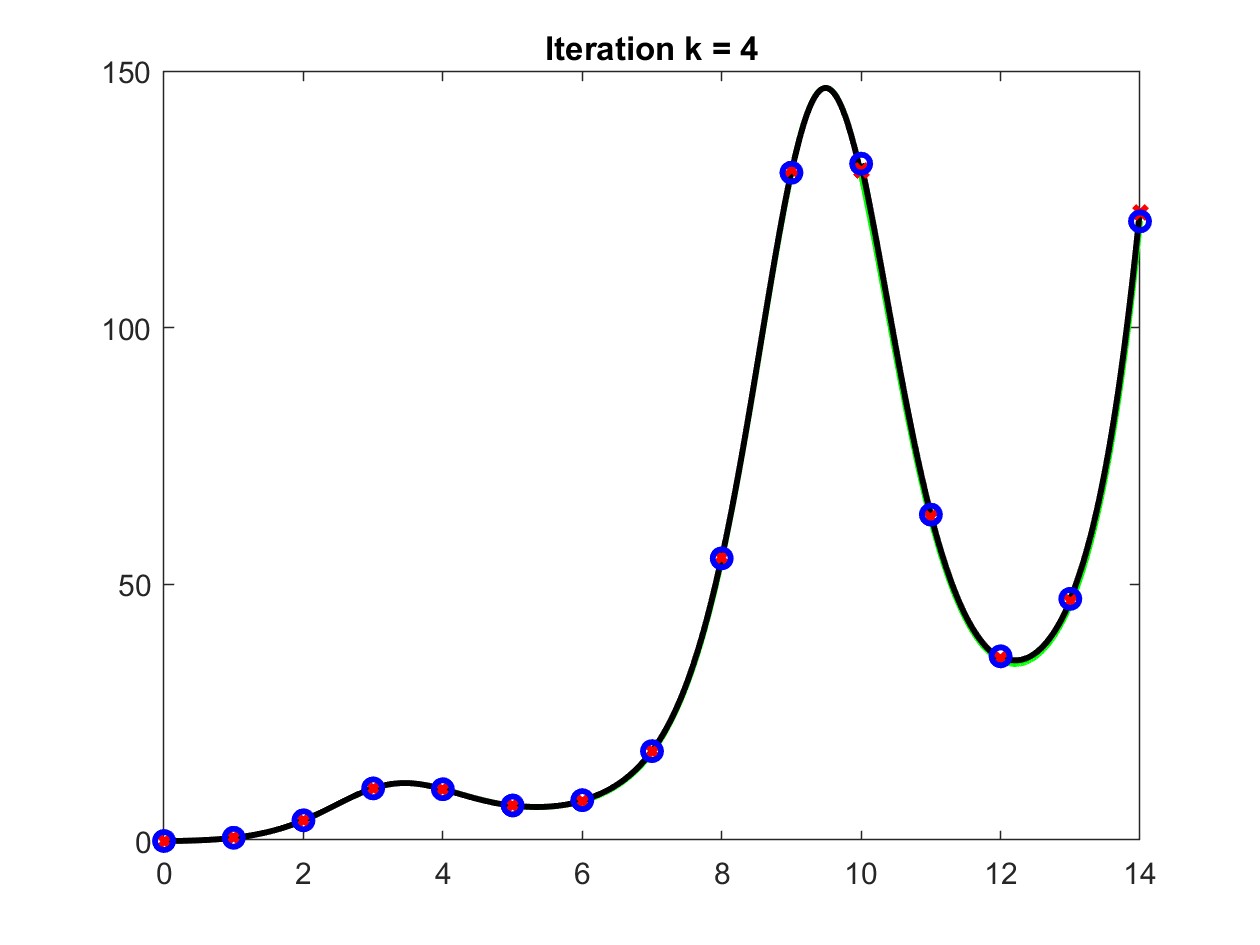
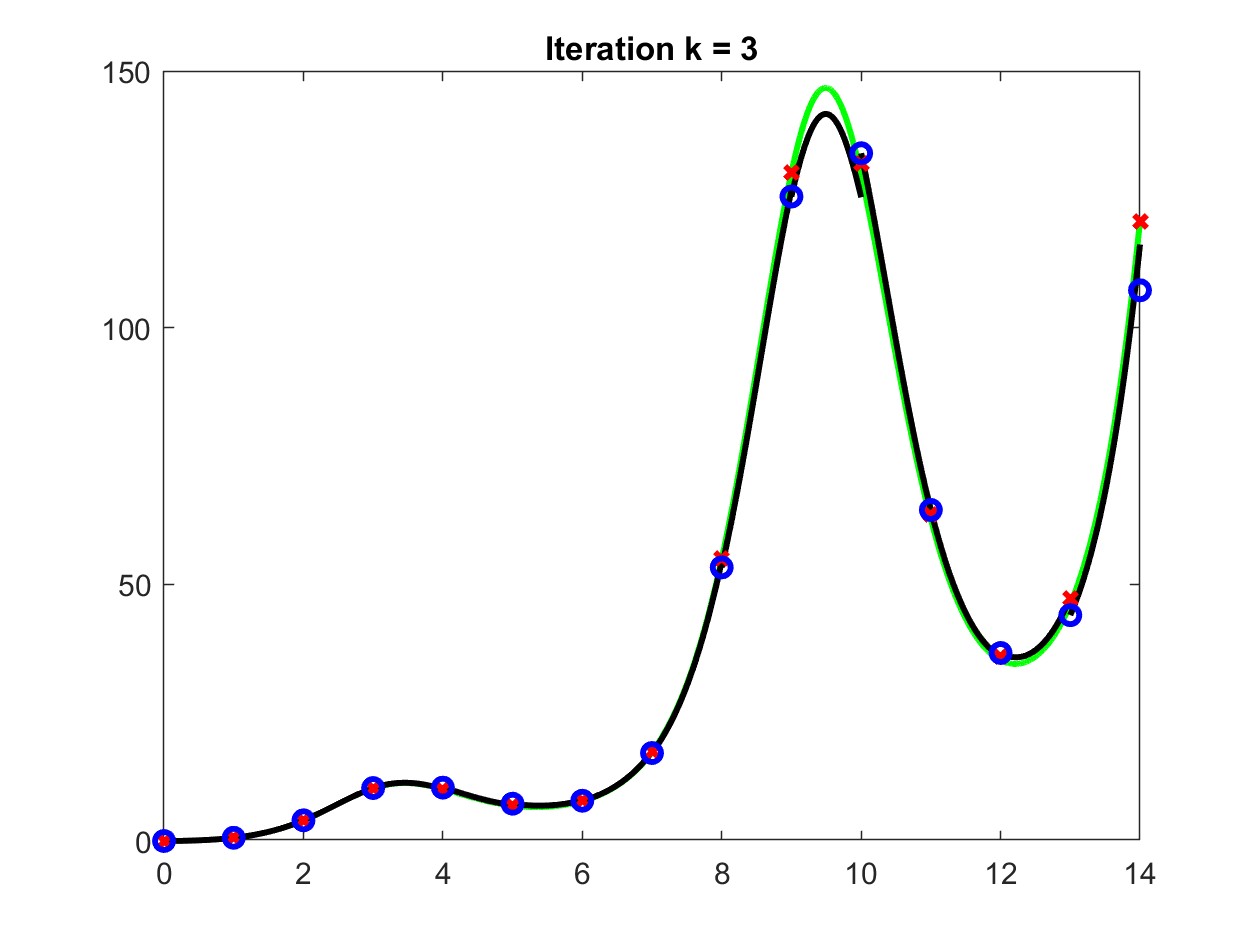
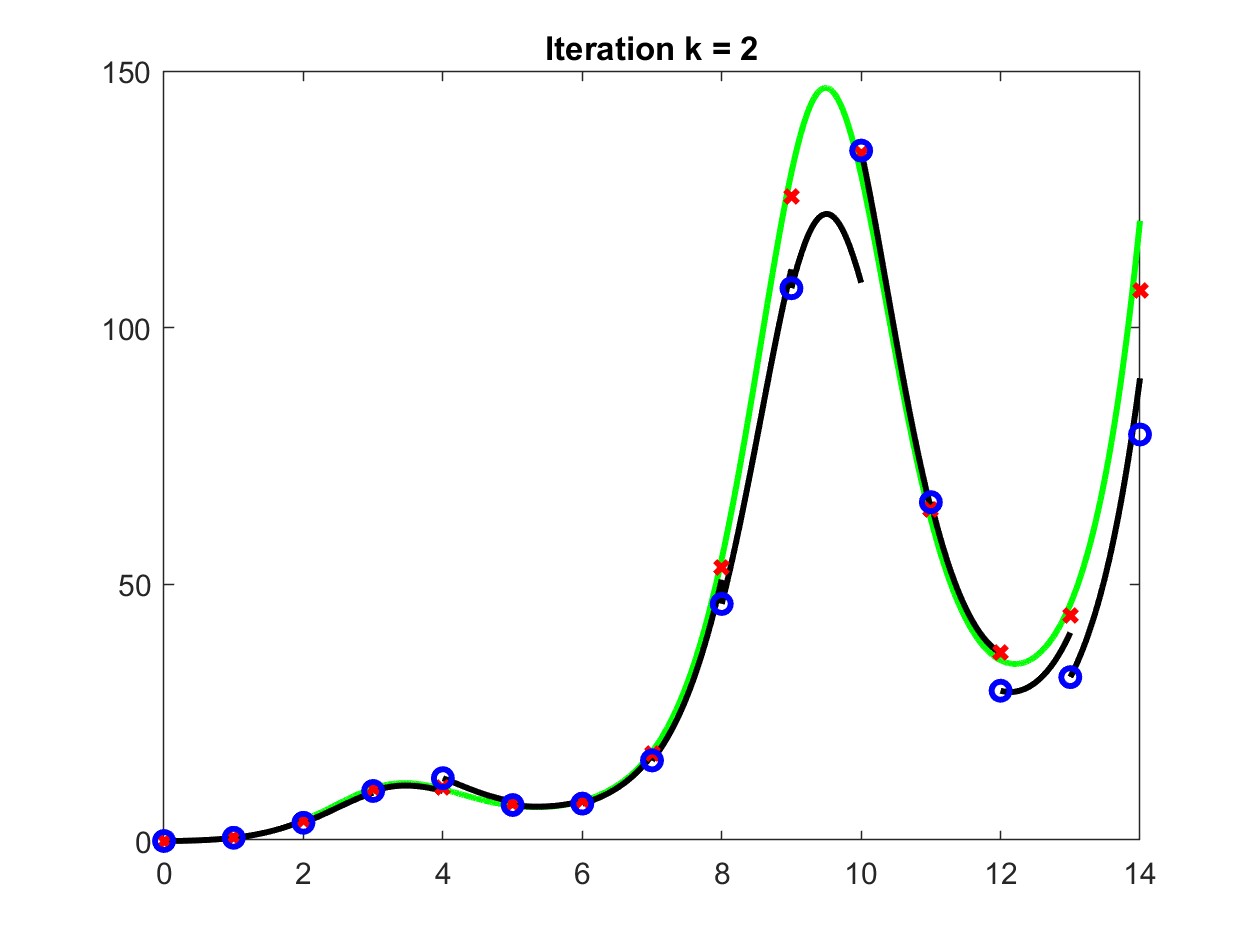
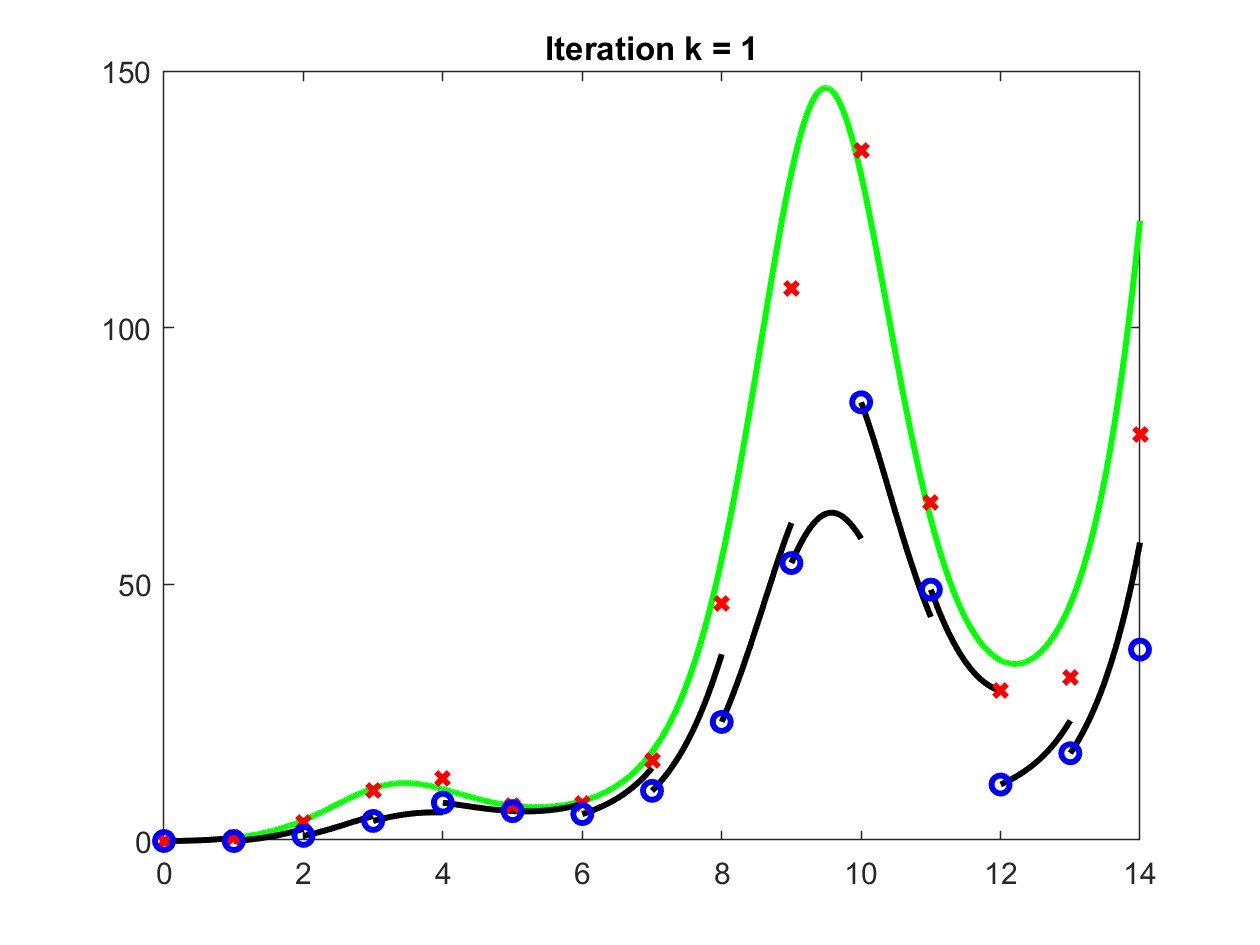
U(n+1) = U(n+1) + u\_fine(n, end) - U\_k(n+1);

end

U\_0 = U

End

**..Some plots now..**

****