Finite-Difference Methods for Dynamic Economies

Graham Candler
Aerospace Engineering and Mechanics
Army HPC Research Center
University of Minnesota

EUI 7th Summer School
Computational Methods for the Study of Dynamic Economies
September 3-13, 1996

Introduction

- "Algorithms based on finite-difference approximations encounter fundamental difficulties." Robert Wilson, *Handbook of Computational Economics, Vol. 1,* 1996.

- However, algorithms based on finite-difference approximations are used throughout computational physics and engineering to solve many different problems.
- Can we apply some of these methods to problems in economics?

- In these lectures, we will consider several dynamic economies and discuss how to solve them using finite-difference methods.

- We will start with a simple approach, and see how that works. Then we will use models equations and simple theory to improve the methods.

- A lot more work needs to be done to tailor the methods for these problems.

Stochastic Dynamic Programming

- Agent's Problem:

$$\max_c E \int_0^\infty e^{-\rho t}\, u(c)\, dt$$
$$\text{s.t.}\quad dk = \big(f(k) - c\big)\, dt + \sigma(k)\, dz, \quad k(0) \text{ given}$$

where:

$$c = \text{consumption}$$
$$k = \text{capital stock}$$
$$z = \text{Brownian motion}$$

- The general dynamic programming equation for the value function, $V(k)$, is:

$$V_t + \rho V = \max_c \Big\{ u(c) + V_k\big(f(k) - c\big) + \tfrac{1}{2}\sigma^2 V_{kk} \Big\}$$

let:

$$u(c) = c^\omega/\omega$$
$$f(k) = \lambda k^\theta - \delta k$$
$$\sigma(k) = \sigma k$$

- The first-order condition, $u'(c) = V_k = c^{\omega-1}$, gives the differential equation:

$$V_t + \rho V = \frac{1-\omega}{\omega} V_k^{\omega/\omega-1} + V_k\big(\lambda k^\theta - \delta k\big) + \tfrac{1}{2}\sigma^2 k^2 V_{kk}$$

Exact Solutions for $\sigma = 0$

- Steady-state solution ($k = k_s$):
  - solve first-order conditions of original problem:

$$\begin{aligned} f'(k_s) &= \rho \\ \theta \lambda k_s^{\theta-1} - \delta &= \rho \end{aligned}$$

  - obtain:

$$\begin{aligned} k_s &= \left(\frac{\rho + \delta}{\lambda \theta} \right)^{1/\theta - 1} \\ c_s &= \lambda k_s^{\theta} - \delta k_s \end{aligned}$$

- Special case where $\theta = \omega = \frac{1}{2}$:

$$\begin{aligned} V &= \frac{\phi^{\omega-1}}{\omega} k^{\omega} + \frac{\lambda}{\rho} \phi^{\omega-1} \\ c &= V_k^{1/\omega-1} = \phi k \\ \phi &= \frac{\rho + \delta \omega}{1 - \omega} \end{aligned}$$

Approach to Computing $V(k)$

- Find time-independent solution: $V_t = 0$
- Either solve the functional equation:

$$\rho V = \frac{1-\omega}{\omega} V_k^{\omega/\omega-1} + V_k\big(\lambda k^\theta - \delta k\big) + \tfrac{1}{2}\sigma^2 k^2 V_{kk}$$

  subject to boundary conditions at $k = 0$ and $k = k_{max}$ (BVP)

- Or solve the time-dependent PDE:

$$V_t + \rho V = \frac{1-\omega}{\omega} V_k^{\omega/\omega-1} + V_k\big(\lambda k^\theta - \delta k\big) + \tfrac{1}{2}\sigma^2 k^2 V_{kk}$$

  for the time-independent solution, with an initial condition and subject to boundary conditions (IBVP)

  - use (guessed) initial condition and integrate in time until solution is no longer a function of the initial condition

Approach to Computing $V(k)$

- Time-dependent approach is better:
  - easier to generalize to multiple states
  - can use theory of PDEs and well-developed numerical methods
  - can use size of time step to regulate the progression of the solution
- Many methods can be used for the solution of this problem:
  - finite-difference, finite-volume, finite-element methods

Finite-Difference Methods

- Approximate the function $V(t,k)$ on an equispaced time / capital-stock grid:

$$V(t,k) = V(t_n, k_i) = V(n\,\Delta t, i\,\Delta k) = V_i^n$$

where

$$n = 0, 1, 2, \ldots \qquad i = 0, 1, 2, \ldots, i_{max}$$

- Use finite-difference approximations to represent the derivatives:

$$V_k(t,k) = \frac{\partial V}{\partial k} = \lim_{\Delta k \to 0} \frac{V(t, k+\Delta k) - V(t,k)}{\Delta k}$$

- Use a Taylor series approximation for $V(t, k+\Delta k)$:

$$\begin{aligned}
V(t, k+\Delta k) &= V(t,k) + \Delta k\, V_k(t,k) + \tfrac{1}{2}\Delta k^2\, V_{kk}(t,k) + \ldots \\
\frac{V(t,k+\Delta k) - V(t,k)}{\Delta k} &= V_k(t,k) + \tfrac{1}{2}\Delta k\, V_{kk}(t,k) + \ldots \\
&= V_k(t,k) + O(\Delta k) \\
\frac{V_{i+1}^n - V_i^n}{\Delta k} &= (V_k)_i^n + O(\Delta k)
\end{aligned}$$

- This is a first-order accurate forward-difference approximation to V_k

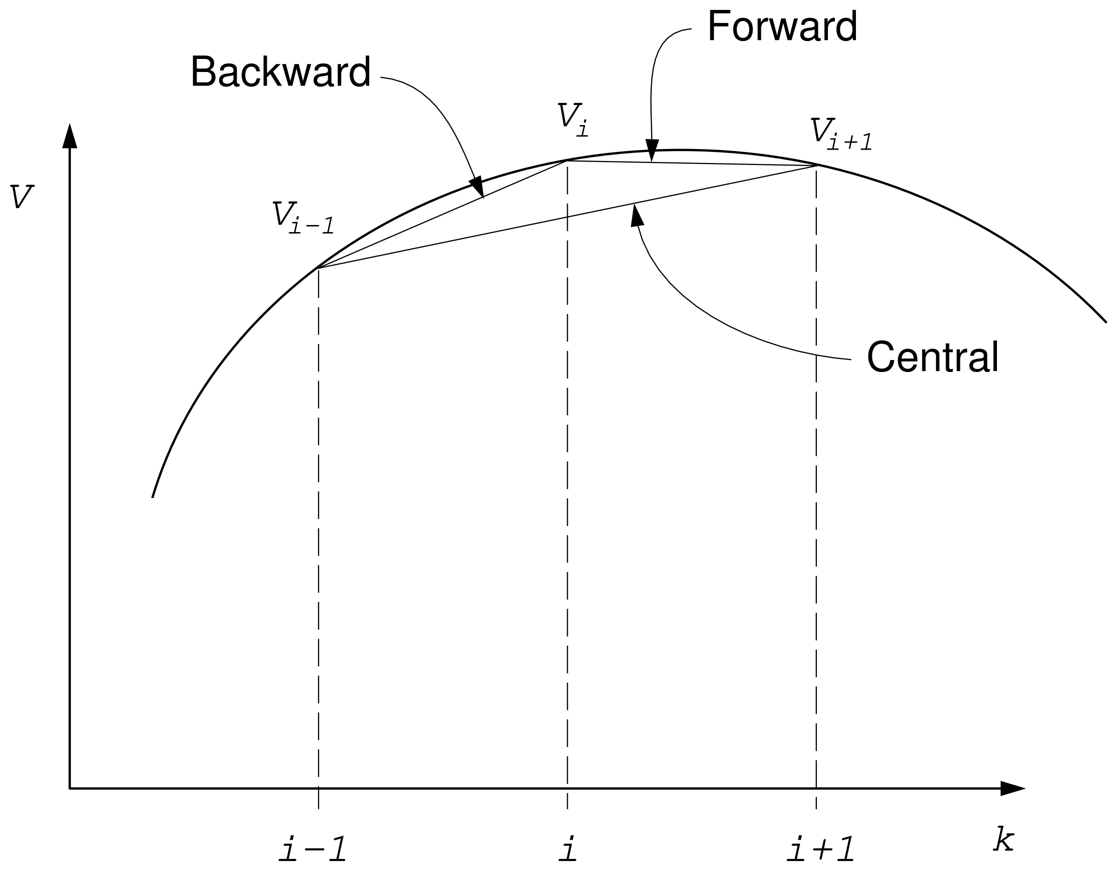
- Similarly, we can derive:

$$(V_k)_i^n = \frac{V_i^n - V_{i-1}^n}{\Delta k} + O(\Delta k) \qquad \text{backward difference}$$
$$(V_k)_i^n = \frac{V_{i+1}^n - V_{i-1}^n}{2\Delta k} + O(\Delta k^2) \qquad \text{central difference}$$
$$(V_k)_i^n = \frac{V_{i+1}^n - V_i^n}{\Delta k} + O(\Delta k) \qquad \text{forward difference}$$

- We can derive approximations of arbitrary order of accuracy by taking linear combinations of Taylor series expansions of $V_{i\pm m}$
  - the more points used, the higher the accuracy
  - central-differences are more accurate than biased approximations
- We can also derive higher-order derivatives. For example

$$(V_{kk})_i^n = \frac{V_{i+1}^n - 2V_i^n + V_{i-1}^n}{\Delta k^2} + O(\Delta k^2) \qquad \text{central difference}$$

Finite-Difference Approximations to $(V_k)_i$



- For this case, the central difference approximation is more accurate than the first-order, one-sided differences.

Finite-Difference Solution of $V(k)$

- Time-dependent differential equation:

$$V_t + \rho V = \frac{1-\omega}{\omega} V_k^{\omega/\omega-1} + V_k\left(\lambda k^\theta - \delta k\right) + \tfrac{1}{2}\sigma^2 k^2 V_{kk}$$

- Use a backward-difference approximation for V_t (for ease of solution; more later):

$$(V_t)_i^n = \frac{V_i^{n+1} - V_i^n}{\Delta t} + O(\Delta t)$$

- Use central-difference approximations for V_k and V_{kk}:

$$\frac{V_i^{n+1} - V_i^n}{\Delta t} + \rho V_i^n = \frac{1-\omega}{\omega}\left(\frac{V_{i+1}^n - V_{i-1}^n}{2\Delta k}\right)^{\omega/\omega-1} + \left(\frac{V_{i+1}^n - V_{i-1}^n}{2\Delta k}\right)\left(\lambda k_i^\theta - \delta k_i\right) + \tfrac{1}{2}\sigma^2 k_i^2 \left(\frac{V_{i+1}^n - 2V_i^n + V_{i-1}^n}{\Delta k^2}\right) + O(\Delta t) + O(\Delta k^2)$$

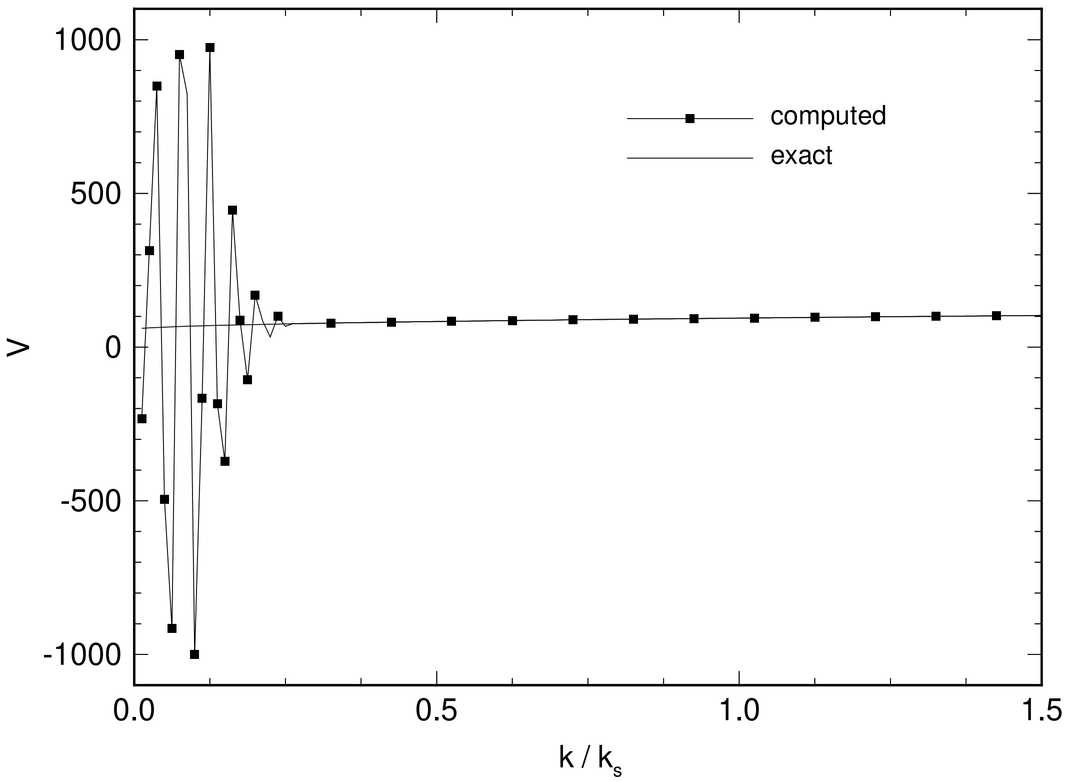
- Solve explicitly for V_i^{n+1}, $\quad i = 0, 1, 2, \ldots, i_{max}$, given an initial guess for V_i^0, and boundary conditions at $k = 0, k_{max}$

Code

- Initialize:
  - make grid: $\Delta k = \frac{k_{max}}{i_{max}}; \quad k_{max} = 2k_s; \quad k_i = i\,\Delta k$
  - set initial values of V_i: say $V_i = A\sqrt{k_i}$
- Solve PDE:
  - compute time step: $\Delta t = \mathrm{CFL}\frac{\Delta k}{a}$ (CFL is a constant, to be discussed later)
  - compute change in solution: $\Delta V_i^n = V_i^{n+1} - V_i^n, \quad \text{for all } i$
  - compute residual (L_2 error norm): $\varepsilon^n = \frac{1}{i_{max}\,\Delta t}\sqrt{\sum_i \left(\Delta V_i^n\right)^2}$
  - update solution: $V_i^{n+1} = V_i^n + \Delta V_i^n$
  - continue until ε^n is small (solution is no longer changing)
- Output the solution

Results Using Central Differences

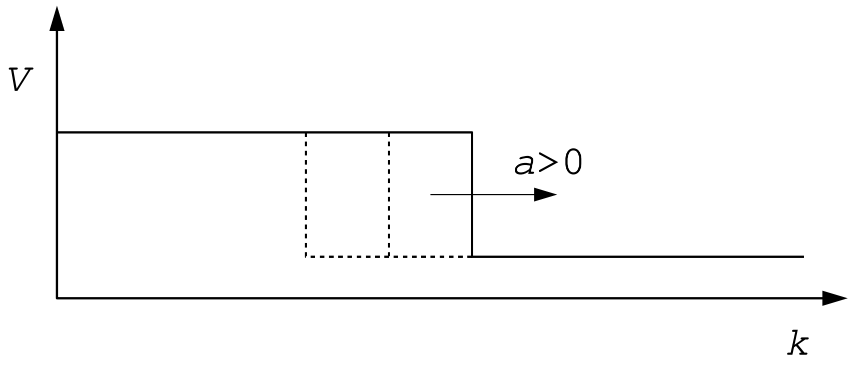
- $\omega = \theta = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$
- Start with the exact solution and run 40 time steps:



- This is not good!

Model Equation: $V_t + aV_k = 0$

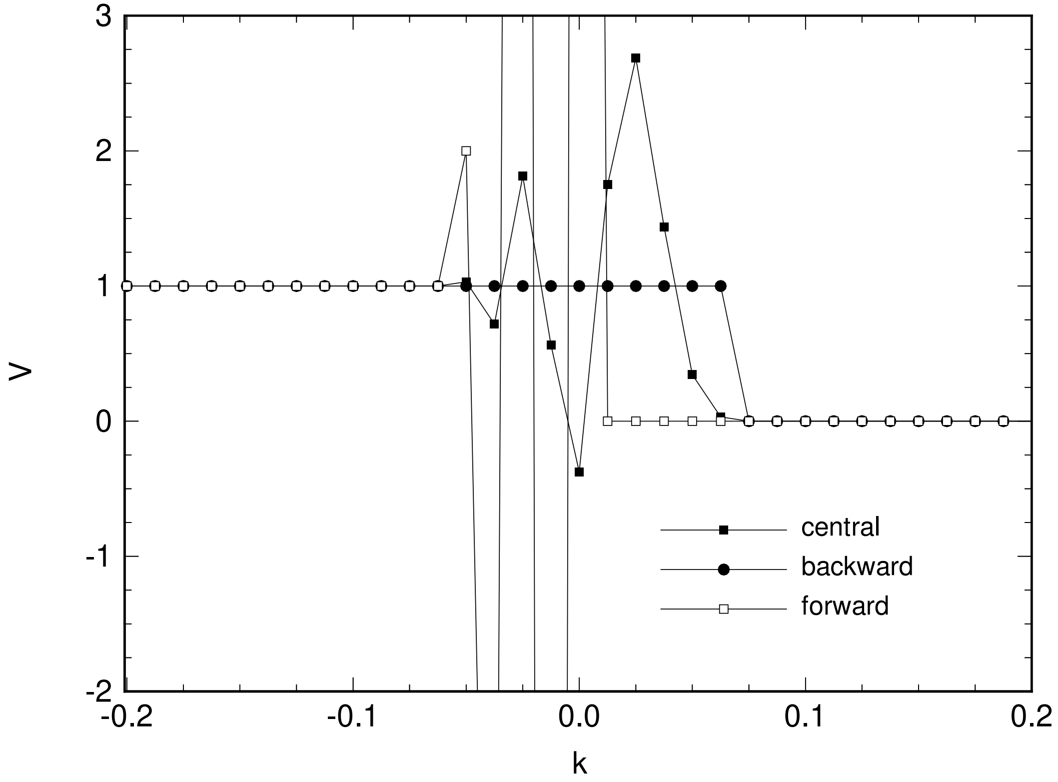
- Note that with $\sigma = 0$, there are only first derivatives in the expression for V_t
- Consider the wave equation: $V_t + aV_k = 0$



- Has an exact solution and captures the key features of the real problem.
- Try solving this problem with the central, backward, and forward difference approximations.

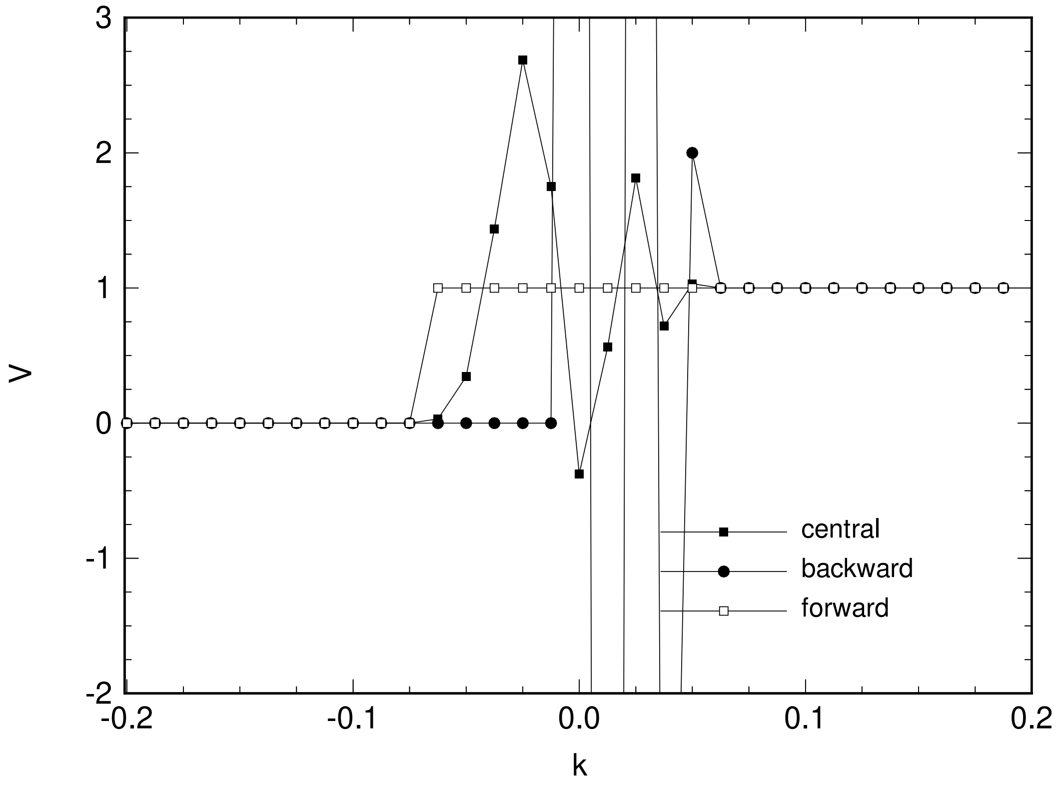
Solution of the Wave Equation ($a > 0$)

- Start with a step function and run 5 time steps:



- For $\Delta t = \frac{\Delta k}{a}$, the backward difference gives the exact solution. Other methods give spurious oscillations (for all choices of Δt).

Solution of the Wave Equation ($a < 0$)



- Now, the forward difference gives the exact solution.
- The trajectory of the solution in space-time determines the differencing scheme.

Von Neumann Stability Analysis

- It is possible to formalize the results of the example discussed above by computing the amplification factor of a finite-difference method.
- Decompose the solution, V_i^n, into the exact solution, $\bar{V}_i^n$, and the error, ϵ_i^n:

$$V_i^n = \bar{V}_i^n + \epsilon_i^n$$

  Substitute into the method, and solve for the evolution of the error.
- Consider the central difference method applied to $V_t + aV_k = 0$, $a > 0$:

$$\frac{V_i^{n+1} - V_i^n}{\Delta t} = -\frac{a}{2\Delta k}\left(V_{i+1}^n - V_{i-1}^n\right)$$

  since $\bar{V}_i^n$ satisfies the equation exactly, we obtain the equation for the errors:

$$\frac{\epsilon_i^{n+1} - \epsilon_i^n}{\Delta t} = -\frac{a}{2\Delta k}\left(\epsilon_{i+1}^n - \epsilon_{i-1}^n\right)$$

- Now substitute a Fourier series representation of the error:

$$\epsilon_i^n = \sum_{j=-N}^{N} E_j^n e^{\imath\, i\phi}$$

  where $\imath = \sqrt{-1}$ and E_j^n is the amplitude of the j^{th} harmonic. $-\pi \leq \phi \leq \pi$ is the phase angle.

Von Neumann Stability Analysis

- Now, we can observe how a single harmonic is amplified by the scheme. Consider the time evolution of $E_j^n e^{\imath i\phi}$:

$$\frac{E^{n+1} - E^n}{\Delta t} e^{\imath i\phi} = -\frac{a}{2\Delta k}\left(E^n e^{\imath (i+1)\phi} - E^n e^{\imath (i-1)\phi}\right)$$
$$E^{n+1} - E^n = -\frac{a\Delta t}{2\Delta k} E^n \left(e^{\imath \phi} - e^{-\imath \phi}\right)$$

- Define the amplification factor, G, as

$$G = \frac{E^{n+1}}{E_n}$$

- The error will not grow over time if

$$|G| = \left|\frac{E^{n+1}}{E_n}\right| \leq 1 \qquad \text{for all } \phi$$

- For this case, we have

$$G = 1 - \imath \frac{a\Delta t}{\Delta k} \sin\phi$$
$$|G|^2 = 1 + \left(\frac{a\Delta t}{\Delta k}\right)^2 \sin^2\phi$$

- Therefore, for non-zero time steps the solution is always unstable. This is an unconditionally unstable scheme.

Von Neumann Stability Analysis

- For the backward difference scheme, the amplification factor is:

$$G = 1 - 2\frac{a\Delta t}{\Delta k}\sin^2\frac{\phi}{2} - \imath\,\frac{a\Delta t}{\Delta k}\sin\phi$$

  and the scheme is stable for

$$0 < \frac{a\Delta t}{\Delta k} \leq 1$$

- This result is commonly known as the Courant-Friedrichs-Lewy or CFL condition. The quantity, $\frac{a\Delta t}{\Delta k}$, is known as the CFL number or the Courant number.
- The backward difference method is said to be conditionally stable.
- Remember that Von Neumann stability analysis only works for linear problems where it is possible to superimpose the Fourier components of the error.
  - Of course, we don't care about linear problems, but the results of the stability analysis are very useful for guiding the design of the methods for nonlinear problems.
  - In practice, nonlinear problems usually have a more restrictive time step limit than that implied by the CFL condition.

Implicit Time Integration Methods

- The CFL condition can be stringent because as the mesh is refined ($\Delta k \to 0$), the time step decreases also. Therefore, many time steps may be required to obtain a converged solution.
- Consider the following differencing scheme for the linear wave equation:

$$\frac{V_i^{n+1} - V_i^n}{\Delta t} = -\frac{a}{2\Delta k}\left(V_{i+1}^{n+1} - V_{i-1}^{n+1}\right)$$

- This is an implicit scheme because the solution at the new time level cannot be obtained explicitly, but rather depends on its neighbors. (We must solve a tridiagonal system of equations.)
- We get something for the effort of solving the system of equations because the amplification factor is given by:

$$G = \frac{1}{1 + \imath \, \frac{a\Delta t}{\Delta k} \sin\phi}$$

  which always satisfies the CFL condition.
- Therefore, the scheme is unconditionally stable for the linear wave equation.
  - in practice the scheme will be stable for only some range of time steps

Upwind Methods

- The foregoing analysis can be used to develop methods that sense the appropriate direction of differencing, and locally adjust the differencing scheme.
- Note that when $a > 0$ in the wave equation, we use a backward difference, and when $a < 0$ we use a forward difference.
- We can develop a scheme that automatically switches the differencing method depending on the local coefficient on V_k. For example if $a = a(k)$:

$$\frac{V_i^{n+1} - V_i^n}{\Delta t} = -\frac{a_i^+}{\Delta k}\left(V_i^n - V_{i-1}^n\right) - \frac{a_i^-}{\Delta k}\left(V_{i+1}^n - V_i^n\right)$$

  where we a_i^+ and a_i^- are determined by sign of a:

$$\begin{aligned} a_i^+ &= \tfrac{1}{2}\left(a_i + |a_i|\right) \\ a_i^- &= \tfrac{1}{2}\left(a_i - |a_i|\right) \end{aligned}$$

- This is an "upwind" method since the differencing scheme uses data from the upwind direction.

Upwind Method for $V(k)$

- Time-dependent differential equation:

$$V_t + \rho V = \frac{1-\omega}{\omega} V_k^{\omega/\omega-1} + V_k\big(\lambda k^\theta - \delta k\big) + \tfrac{1}{2}\sigma^2 k^2 V_{kk}$$

- Group terms to get the coefficient on V_k:

$$V_t + \rho V = \left(\frac{1-\omega}{\omega} V_k^{1/\omega-1} + \lambda k^\theta - \delta k\right) V_k + \tfrac{1}{2}\sigma^2 k^2 V_{kk}$$

- Obtain the upwind difference method:

$$\begin{aligned}\frac{V_i^{n+1} - V_i^n}{\Delta t} + \rho V_i^n = & -\frac{a_i^+}{\Delta k}\big(V_i^n - V_{i-1}^n\big) - \frac{a_i^-}{\Delta k}\big(V_{i+1}^n - V_i^n\big) \\ & + \tfrac{1}{2}\sigma^2 k_i^2 \left(\frac{V_{i+1}^n - 2V_i^n + V_{i-1}^n}{\Delta k^2}\right) + O(\Delta t) + O(\Delta k)\end{aligned}$$

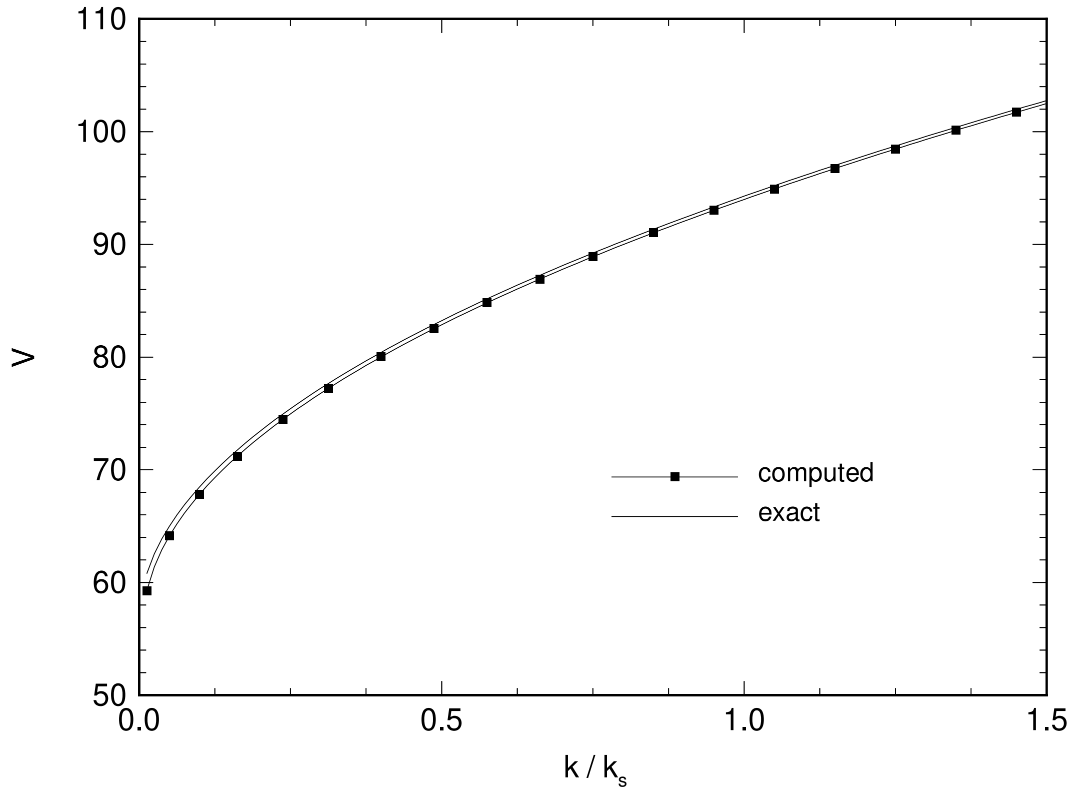
where

$$a_i = -\frac{1-\omega}{\omega}(V_k)_i^{1/\omega-1} - \lambda k_i^\theta + \delta k_i$$

- This is an extension of the upwind approach for a coefficient that depends on V_k.
- Again we can solve explicitly for V_i^{n+1} given an initial guess for V_i^0.

Results Using Upwind Method

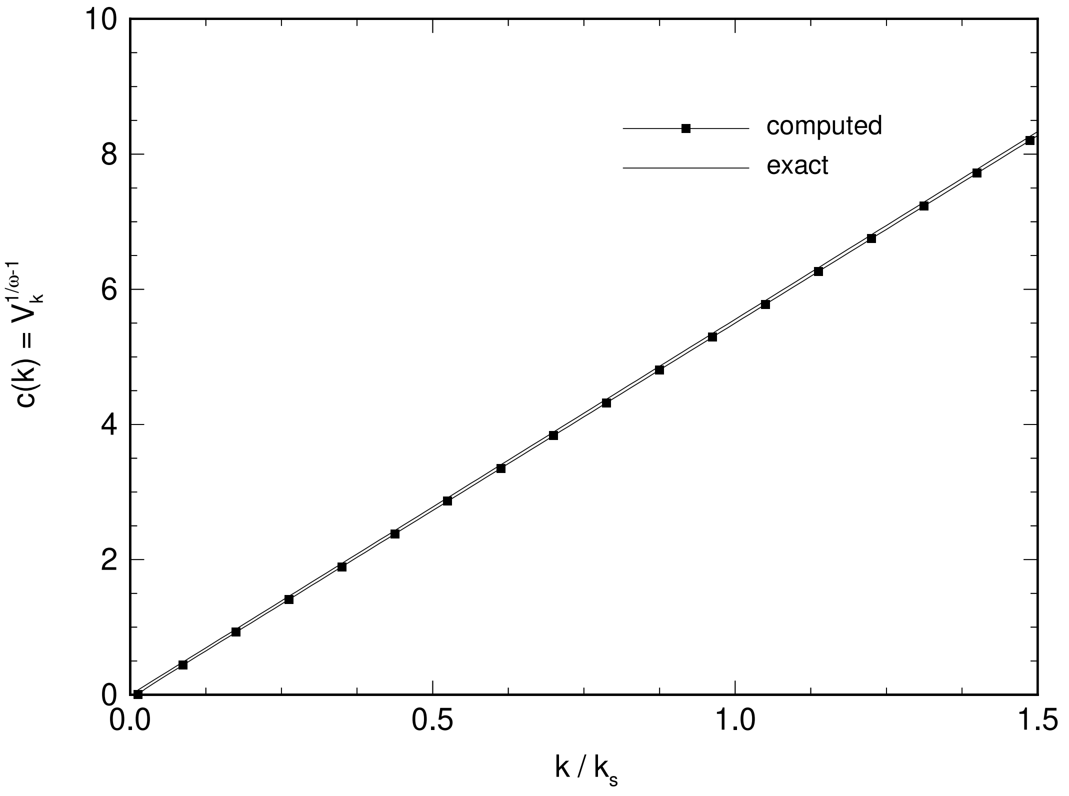
- $\omega = \theta = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$
- Start with a guessed solution and run until error is small:



- This is good!

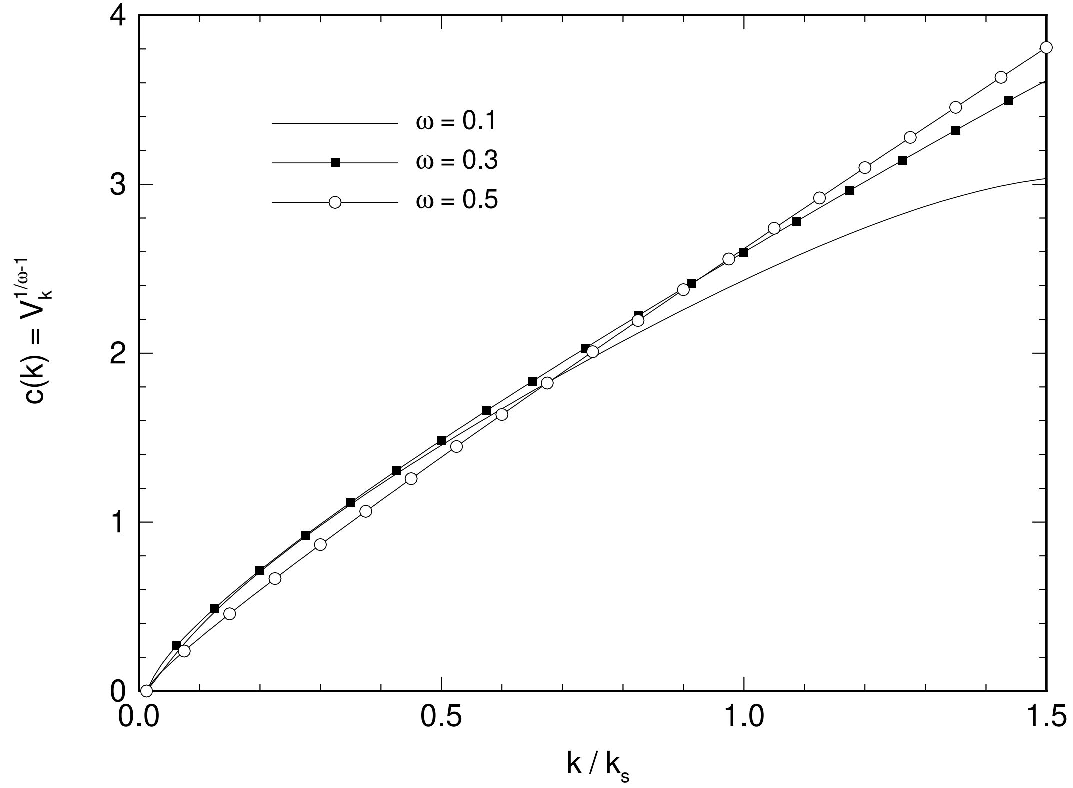
Results Using Upwind Method

- $\omega = \theta = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$
- Consumption function:



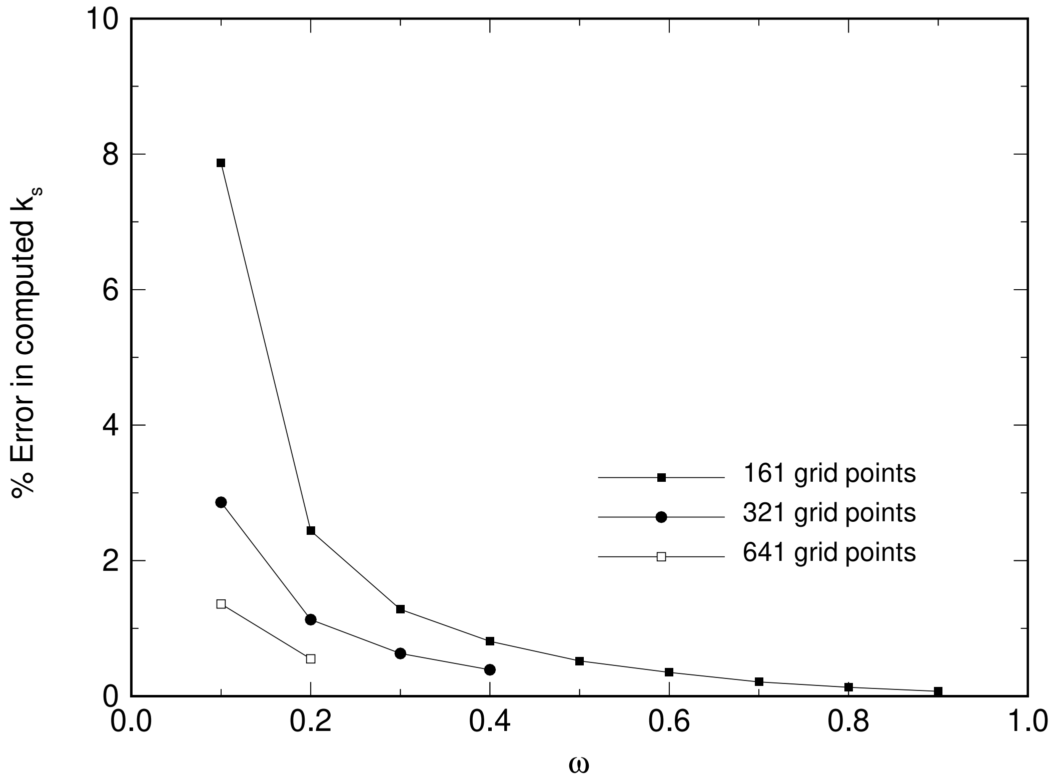
Results Using Upwind Method: ω Variation

- $\theta = 0.4$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$
- Consumption function for various values of ω:



Results Using Upwind Method: Grid Convergence

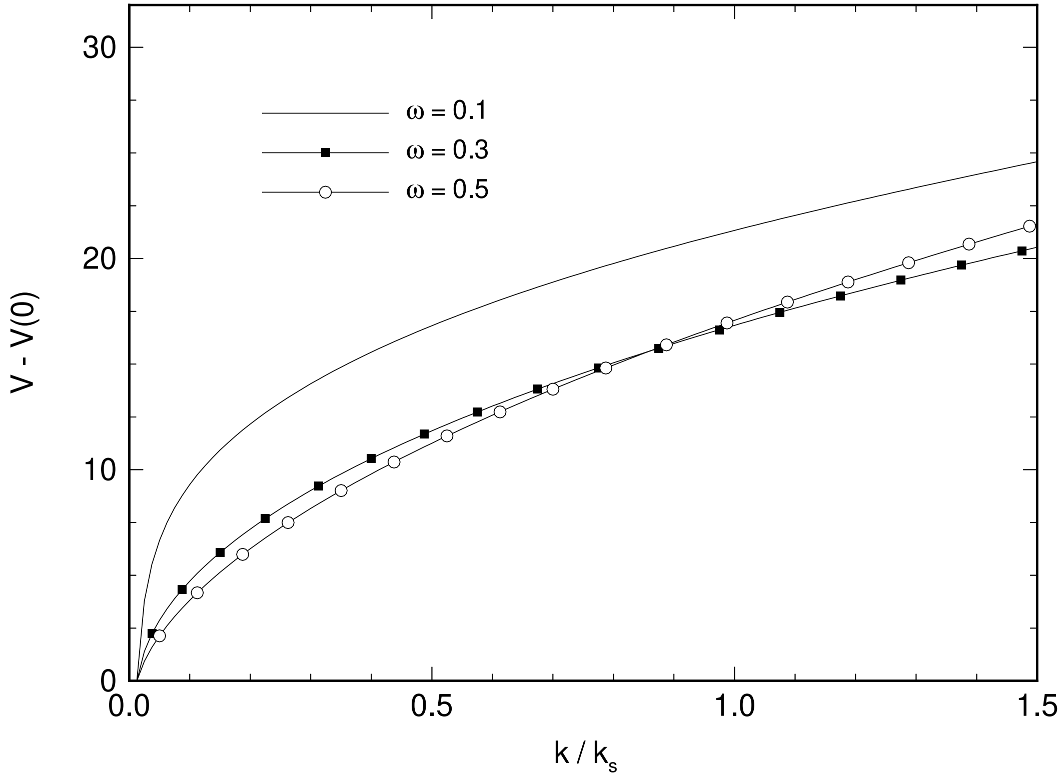
- $\theta = 0.4$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$
- Compare computed steady-state capital stock with exact result:



- Increased grid resolution improves solution.

Results Using Upwind Method: Grid Convergence

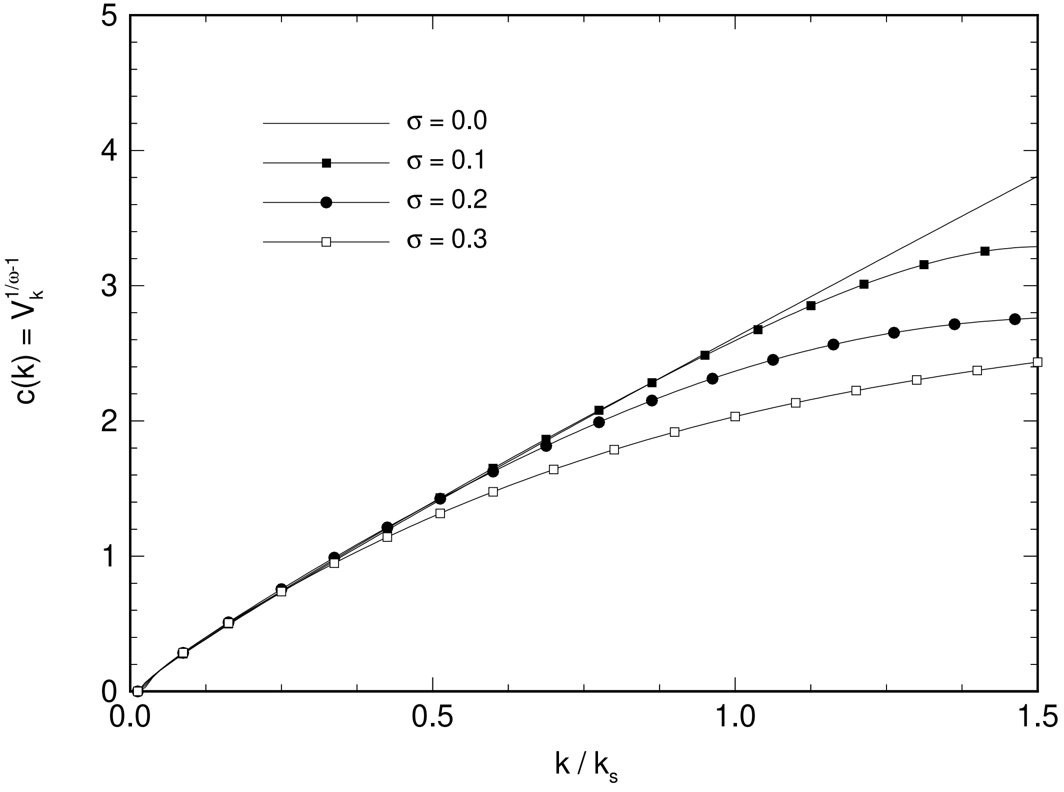
- $\theta = 0.4$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$
- Nonlinearity of $V(k)$ increases with smaller values of ω:



- Increased error is a result of increased nonlinearity.
  - error term in first-order differencing scheme is: $\frac{1}{2}\Delta k V_{kk} = O(\Delta k)$

Results Using Upwind Method: σ Variation

- $\theta = 0.4$, $\omega = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$



Implicit Method for $V(k)$

- Since we are integrating the PDE from a guessed initial condition to the final answer, if we can take larger time steps, we should be able to get to the answer faster.
- We discussed an implicit method above and saw that for the wave equation, it is unconditionally stable.
    - of course, that was for a linear problem
- Let's try developing an implicit method for this problem:
    - use the same upwind differencing method for the k derivatives
    - evaluate the derivatives at the future $(n+1)$ time level

$$\frac{V_i^{n+1} - V_i^n}{\Delta t} + \rho V_i^{n+1} = -\frac{a_i^+}{\Delta k}\left(V_i^{n+1} - V_{i-1}^{n+1}\right) - \frac{a_i^-}{\Delta k}\left(V_{i+1}^{n+1} - V_i^{n+1}\right)$$
$$+ \tfrac{1}{2}\sigma^2 k_i^2 \left(\frac{V_{i+1}^{n+1} - 2V_i^{n+1} + V_{i-1}^{n+1}}{\Delta k^2}\right) + O(\Delta t) + O(\Delta k)$$

for simplicity, use $a_i^\pm$ at the present (n) time level

Implicit Method for $V(k)$

- If we let $\Delta V_i^n = V_i^{n+1} - V_i^n$, we can write the implicit problem as:

$$\begin{aligned}\big(\alpha_i^- - \beta_i\big)\Delta V_{i+1}^n + \big(1 + \alpha_i^+ - \alpha_i^- + 2\beta_i + \Delta t\rho\big)\Delta V_i^n + \big(-\alpha_i^+ - \beta_i\big)\Delta V_{i-1}^n = \\ -\alpha_i^+\big(V_i^n - V_{i-1}^n\big) - \alpha_i^-\big(V_{i+1}^n - V_i^n\big) + \beta_i\big(V_{i+1}^n - 2V_i^n + V_{i-1}\big) - \Delta t\rho V_i^n\end{aligned}$$

  where $\alpha^\pm = \frac{\Delta t\, a^\pm}{\Delta k}$ and $\beta = \frac{\Delta t\, \sigma^2 k^2}{2\Delta k^2}$
- This is a tridiagonal system for ΔV_i^n:
  - again, the solution cannot be explicitly computed
  - tridiagonal system is easy and efficient to solve
- Symbolically, we have:

$$\begin{pmatrix} B & A & C & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & B_i & A_i & C_i & & \\ & & & \ddots & \ddots & \ddots & \\ & & & & B & A & C \end{pmatrix}^n \begin{pmatrix} \vdots \\ \Delta V_{i-1} \\ \Delta V_i \\ \Delta V_{i+1} \\ \vdots \end{pmatrix}^n = \begin{pmatrix} \vdots \\ RHS_i \\ \vdots \end{pmatrix}^n$$

Other Implicit Methods for $V(k)$

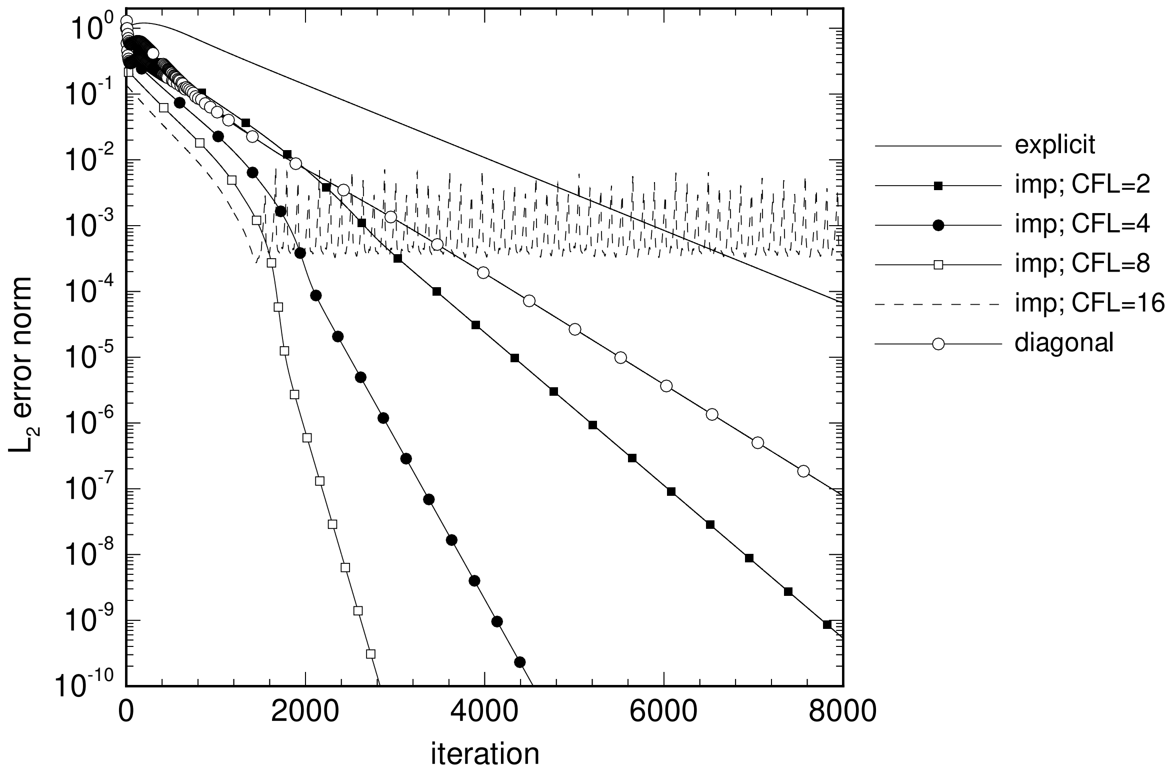
- Many other implicit methods can be derived.
  - not really needed for this one-dimensional problem, but can be useful for higher dimensions.
- Answer is independent of implicit operator.
- Note that the system is diagonally dominant for any Δt:
  - off-diagonal terms may be moved to the *RHS*
  - may be lagged in iteration number or even ignored
  - a direct result of the upwind difference method
- A “diagonal” method may be derived by ignoring off-diagonal terms:

$$
\begin{aligned}
&\big(1 + \alpha_i^+ - \alpha_i^- + 2\beta_i + \Delta t \rho\big)\Delta V_i^n = \\
&\quad - \alpha_i^+\big(V_i^n - V_{i-1}^n\big) - \alpha_i^-\big(V_{i+1}^n - V_i^n\big) + \beta_i\big(V_{i+1}^n - 2V_i^n + V_{i-1}\big) - \Delta t \rho V_i^n
\end{aligned}
$$

  - trivial to solve
  - ignores off-diagonal coupling in implicit operator
  - if $\Delta t \to \infty$, becomes a relaxation method
  - generalizes to multiple dimensions

Comparison of Convergence Rates: L_2 Error Norm

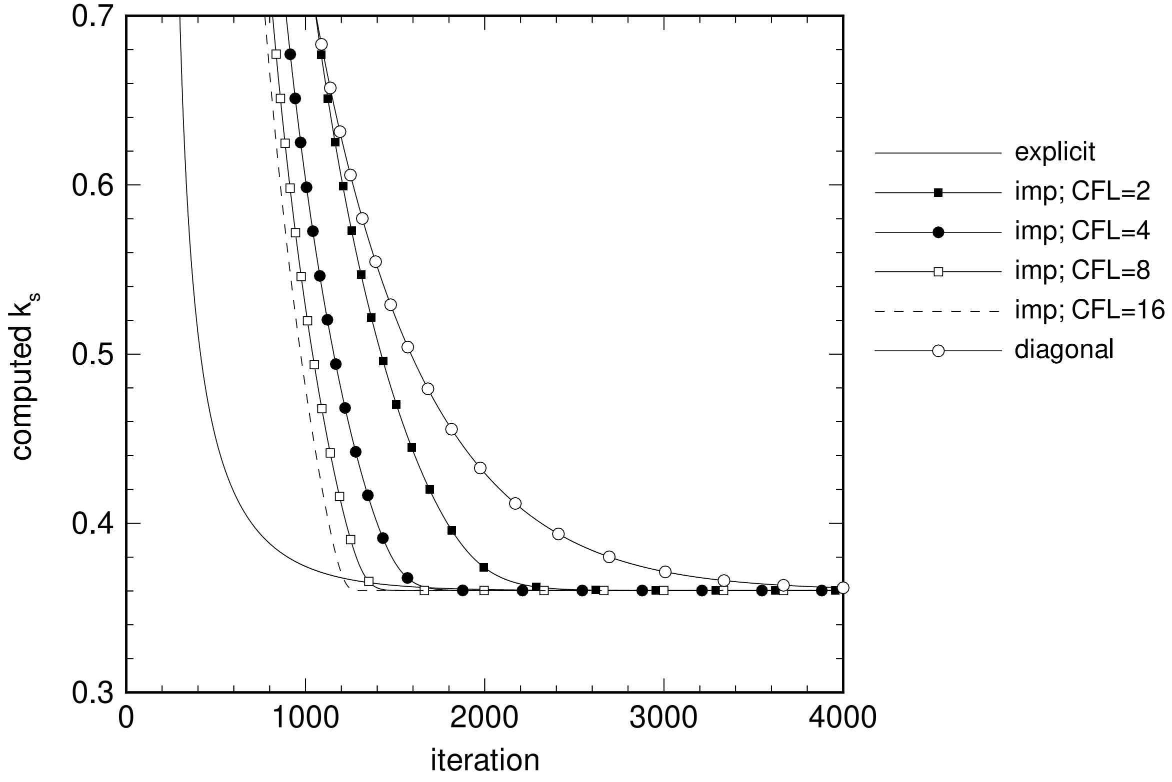
- $\theta = 0.4$, $\omega = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$



- The implicit method is best, but there is an optimal CFL.
- The diagonal method is slower to converge, but is stable for any time step.
- No accounting for cost has been made here.

Comparison of Convergence Rates: k_s

- $\theta = 0.4$, $\omega = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$



- By this measure, the diagonal method is the slowest.

Convergence Rates

- Even for the implicit method, about 2000 iterations are required to obtain a steady-state solution.
  - a very poor initial guess for V was used
  - these PDEs are different than those in physics: $V_k^{\omega/\omega-1}$
  - non-uniform grid spacing would reduce the cost significantly (cluster grid points near regions of strong nonlinearity).
  - coefficient on V_k could be linearized and included in the implicit solution
- Convergence is not all that much different for difficult nonlinear problems in computational physics.
- It is is likely that with more work better convergence rates could be obtained.
- In any case, the solutions take a few seconds on a workstation.

Upwind Method for $c(k)$

- When $\sigma = 0$, derive the time-dependent differential equation:

$$c_t = \frac{u'(c)}{u''(c)}\big(f'(k) - \rho\big) + \big(f(k) - c(k)\big)c'(k)$$

with $u(c) = c^\omega/\omega$ and $f(k) = \lambda k^\theta - \delta k$

- Obtain:

$$c_t = \big(\lambda k^\theta - \delta k - c\big)c_k + \frac{1}{\omega - 1}\big(\lambda\theta k^{\theta-1} - \delta - \rho\big)c$$

- Upwind finite-difference method:

$$\frac{c_i^{n+1} - c_i^n}{\Delta t} = -\frac{a_i^+}{\Delta k}\big(c_i^n - c_{i-1}^n\big) - \frac{a_i^-}{\Delta k}\big(c_{i+1}^n - c_i^n\big) + S_i^n$$

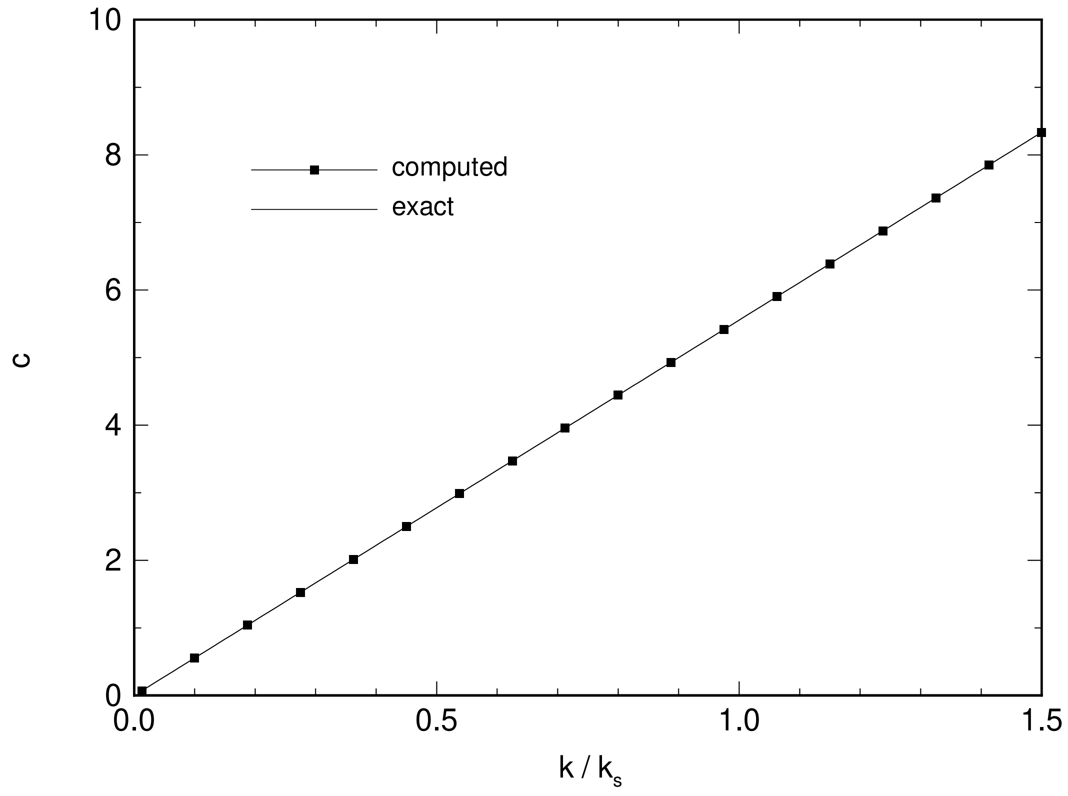
where

$$a_i = -\lambda k_i^\theta + \delta k_i + c_i$$

$$S_i = \frac{1}{\omega - 1}\big(\lambda\theta k_i^{\theta-1} - \delta - \rho\big)c_i$$

Results Using Upwind Method

- $\omega = \theta = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0$



- This is good!

Implicit Method for $c(k)$

- Again evaluate all quantities at future time level:

$$\frac{c_i^{n+1} - c_i^n}{\Delta t} = -\frac{a_i^+}{\Delta k}\left(c_i^{n+1} - c_{i-1}^{n+1}\right) - \frac{a_i^-}{\Delta k}\left(c_{i+1}^{n+1} - c_i^{n+1}\right) + S_i^{n+1}$$

- Linearize the source term:

$$S_i^{n+1} = S_i^n + \left.\frac{\partial S}{\partial c}\right|_i^n \left(c_i^{n+1} - c_i^n\right) + O(\Delta t)$$

- Obtain:

$$\alpha_i^- \Delta c_{i+1}^n + \left(1 + \alpha_i^+ - \alpha_i^- - \Delta t \frac{\partial S}{\partial c}\right) \Delta c_i^n - \alpha_i^+ \Delta c_{i-1}^n = \\ - \alpha_i^+ \left(c_i^n - c_{i-1}^n\right) - \alpha_i^- \left(c_{i+1}^n - c_i^n\right) + \Delta t S_i^n$$

- The convergence rate is poor: $\mathrm{CFL}_{max} \simeq 2$

Model Equation for $c(k)$

- Consider the case where c_k is small. Then:

$$c_t = \frac{1}{\omega - 1}\left(\lambda\theta k^{\theta-1} - \delta - \rho\right)c = sc = S$$

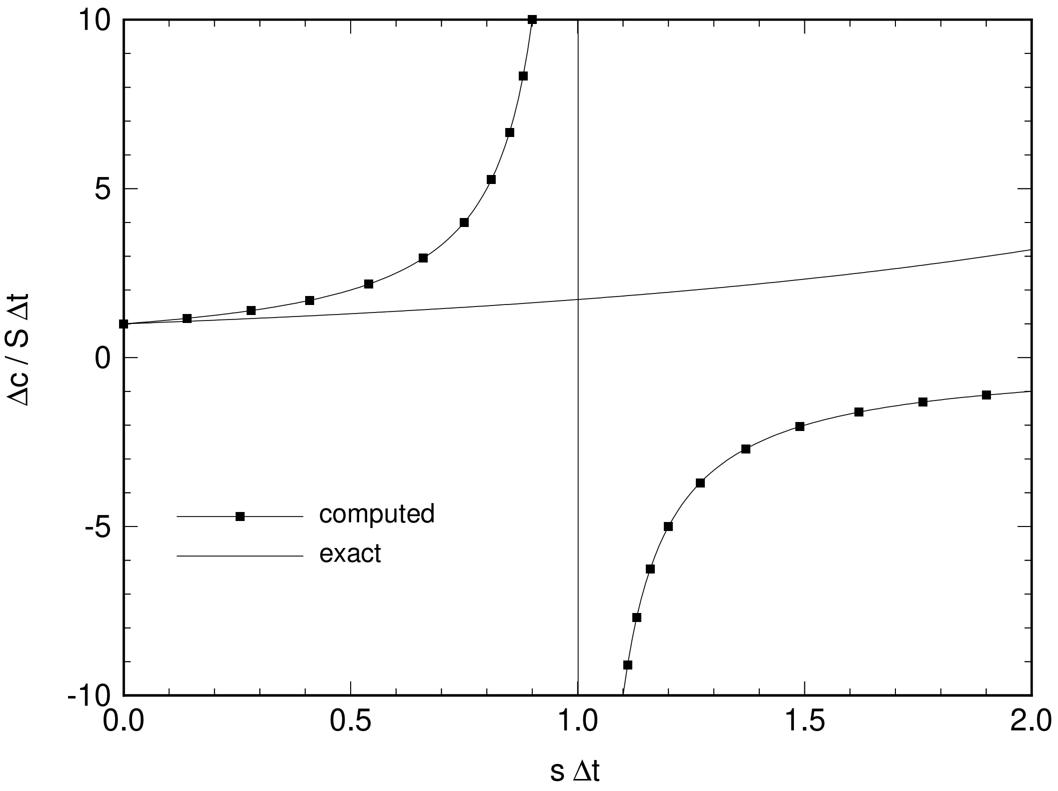
- An implicit representation:

$$\frac{c_i^{n+1} - c_i^n}{\Delta t} = sc^{n+1} \simeq S^n + s\left(c_i^{n+1} - c_i^n\right)$$
$$c^{n+1} = c^n + \frac{\Delta t S^n}{1 - s\Delta t}$$

- If $s > 0$ and Δt is large, can have problems.

Model Equation for $c(k)$

- For constant s, we can compare to exact solution:

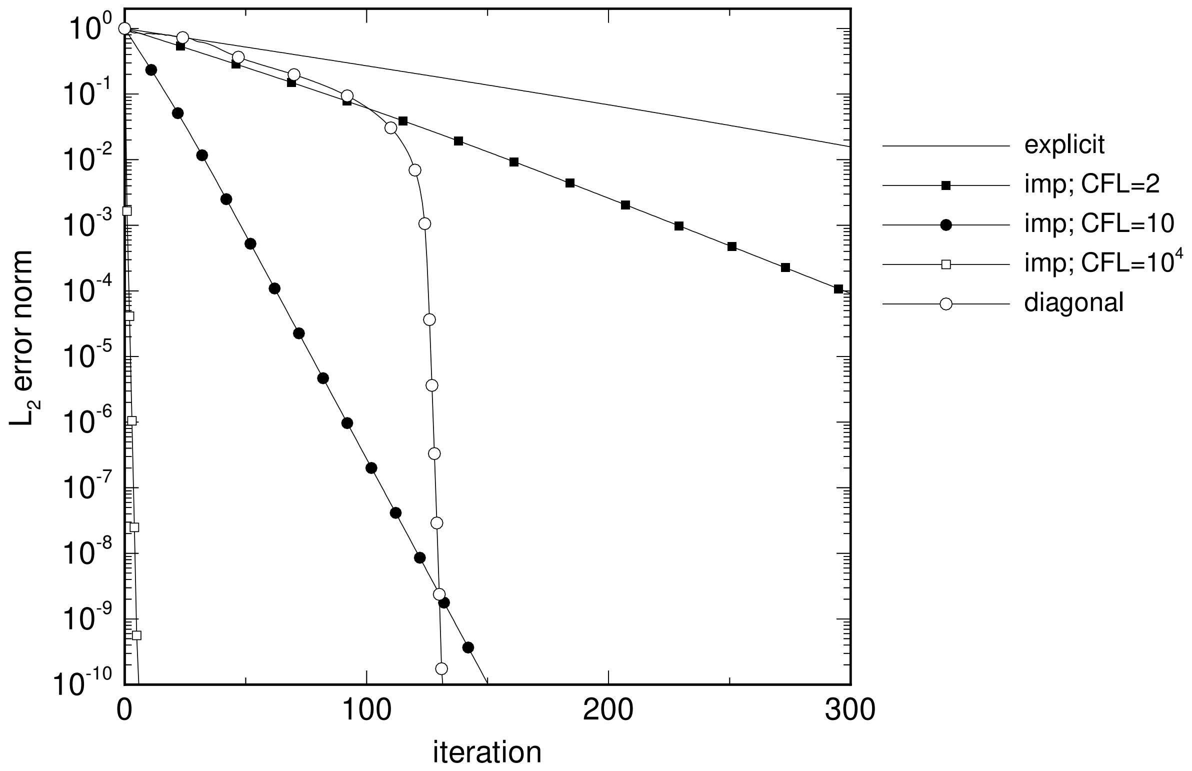


- Clearly, this approach has problems for $s\Delta t$ large.

Comparison of Convergence Rates: L_2 Error Norm

- $\theta = 0.4$, $\omega = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0$
- Let

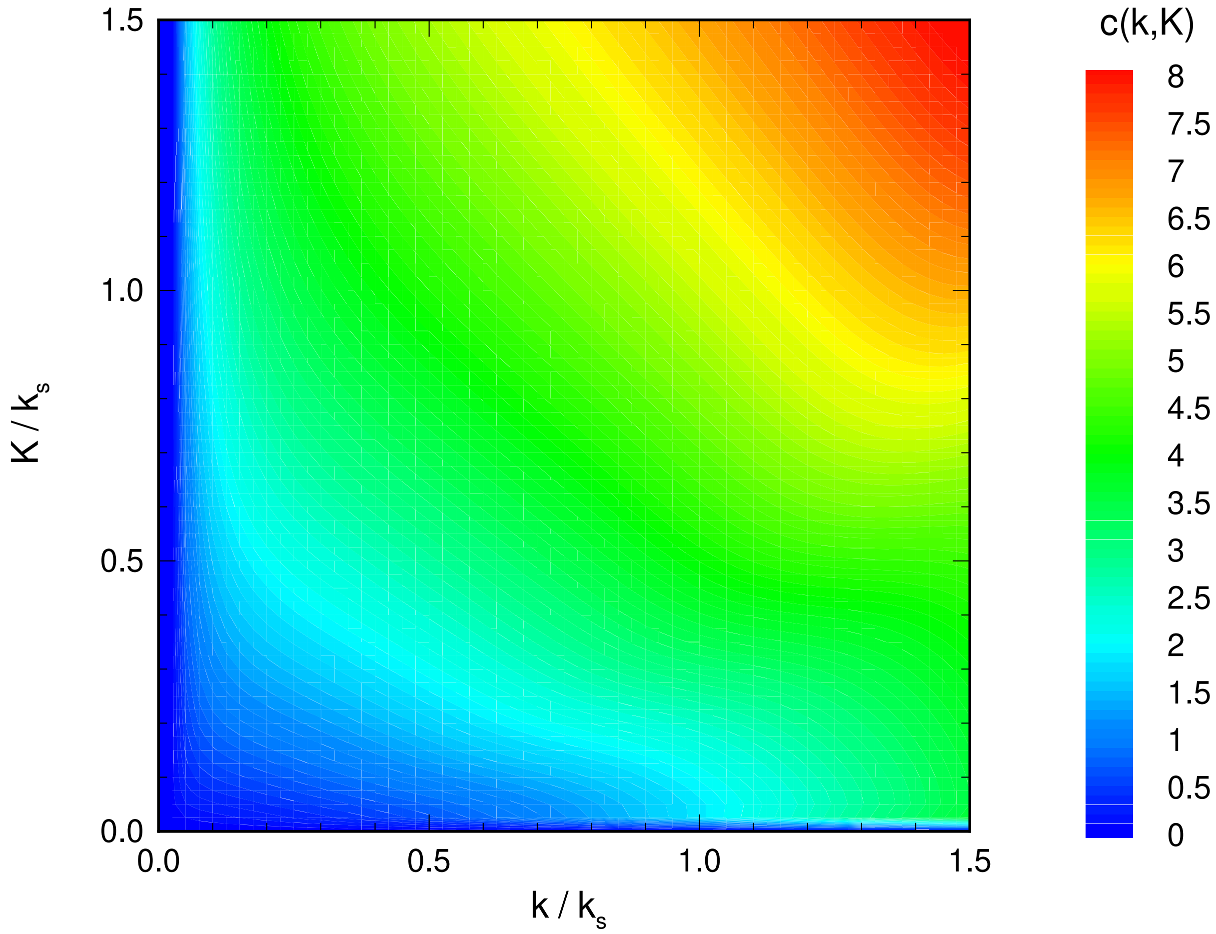
$$\frac{\partial S}{\partial c} \sim \frac{1}{\omega - 1} \lambda \theta k^{\theta - 1} < 0$$



- For this case, the implicit methods are much better.

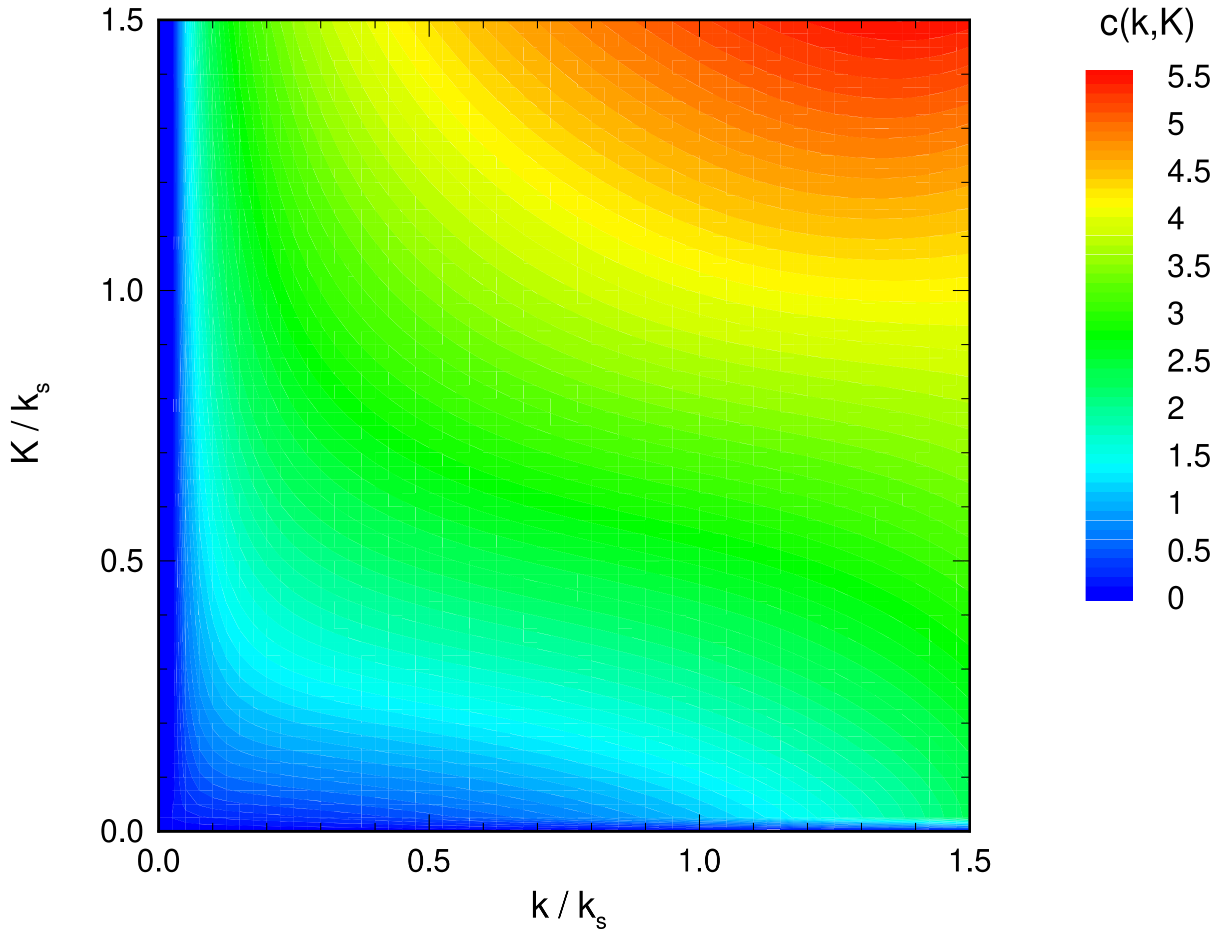
Results Using 2-D Upwind Method: $V(k, K)$

- $\theta = 0.5$, $\omega = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0$, $\tau = 0.0$



Results Using 2-D Upwind Method: $V(k, K)$

- $\theta = 0.5$, $\omega = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0$, $\tau = 0.4$



Stochastic Dynamic Programming (2-D)

- Agent's Problem:

$$\max_c E \int_0^\infty e^{-\rho t}\, u(c)\, dt$$

$$\text{s.t.} \quad dk = \big((1-\tau)r(K)k + w(K) - c + T(K)\big)dt + \sigma(k,K)dz$$

$$dK = \big(F(K) - C(K) - G(K)\big)dt + \sigma(K,K)dz$$

where:

$$\begin{aligned} c &= \text{individual consumption} \\ k &= \text{individual capital stock} \\ C(K) &= \text{aggregate consumption} \\ K &= \text{aggregate capital stock} \\ F(K) &= \text{aggregate output} \\ G(K) &= \text{government spending} \\ T(K) &= \text{transfers} \end{aligned}$$

let:

$$u(c) = c^\omega/\omega, \quad F(k) = \lambda k^\theta - \delta k, \quad r = F_K, \quad w = F - F_K K, \quad G = \tau F_K K$$

Stochastic Dynamic Programming (2-D)

- The general dynamic programming equation for the value function, $V(k, K)$, is:

$$
\begin{aligned}
V_t + \rho V = \max_c \Big\{ u(c) &+ V_k\big((1-\tau)(\theta\lambda K^{\theta-1} - \delta)k + \lambda K^\theta(1-\theta) - c\big) \\
&+ V_K\big(\lambda K^\theta - C(K) - \tau(\theta\lambda K^\theta - \delta K)\big) \\
&+ \tfrac{1}{2}\big(\sigma(k,K)^2 V_{kk} + 2\sigma(k,K)\sigma(K,K)V_{kK} + \sigma(K,K)^2 V_{KK}\big)\Big\}
\end{aligned}
$$

- The first-order condition, $u'(c) = V_k(k, K)$, gives the differential equation:

$$
\begin{aligned}
V_t + \rho V = \frac{1-\omega}{\omega} V_k^{\omega/\omega-1} &+ V_k\big((1-\tau)(\theta\lambda K^{\theta-1} - \delta)k + \lambda K^\theta(1-\theta)\big) \\
&+ V_K\big(\lambda K^\theta - \delta K - V_k(K,K)^{1/\omega-1} - \tau(\theta\lambda K^\theta - \delta K)\big) \\
&+ \tfrac{1}{2}\big(\sigma(k,K)^2 V_{kk} + 2\sigma(k,K)\sigma(K,K)V_{kK} + \sigma(K,K)^2 V_{KK}\big)
\end{aligned}
$$

Finite-Difference Solution of $V(k, K)$

- Now let:

$$V(t,k,K) = V(t_n, k_i, K_j) = V(n\,\Delta t, i\,\Delta k, j\,\Delta K) = V^n_{i,j}$$

- Derive the upwind difference method:

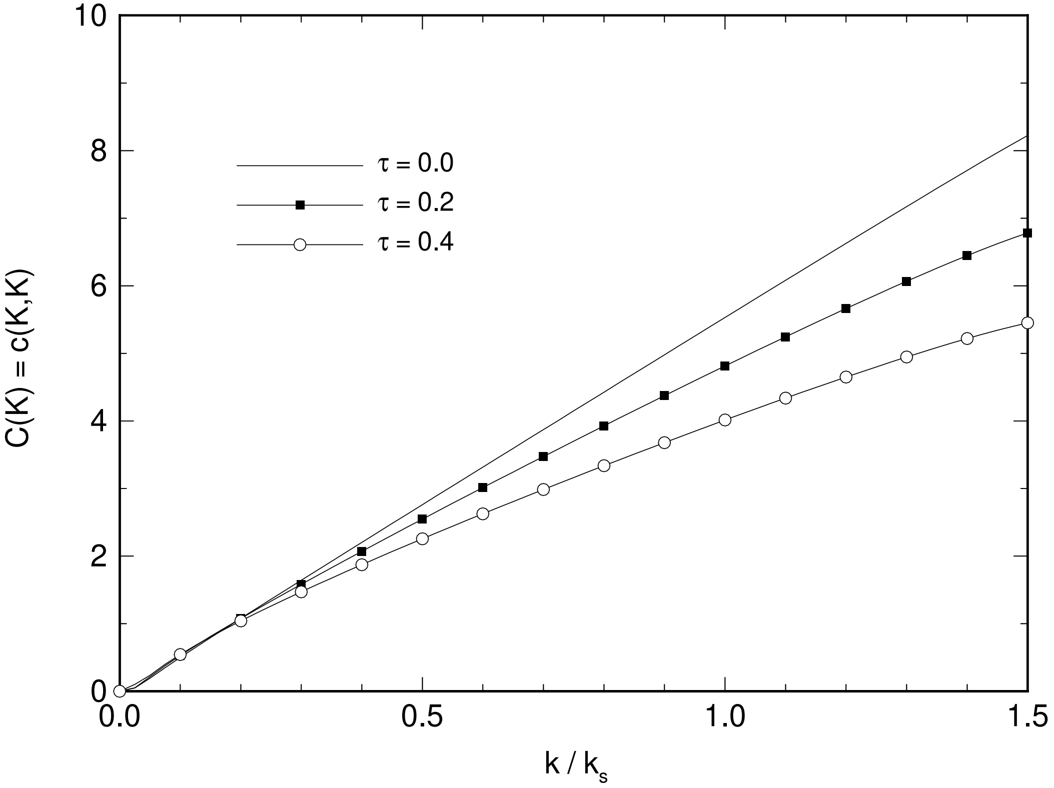
$$\begin{aligned}\frac{V^{n+1}_{i,j} - V^n_{i,j}}{\Delta t} + \rho V^n_{i,j} = &- \frac{a^+_{i,j}}{\Delta k}\left(V^n_{i,j} - V^n_{i-1,j}\right) - \frac{a^-_{i,j}}{\Delta k}\left(V^n_{i+1,j} - V^n_{i,j}\right) \\ &- \frac{b^+_{i,j}}{\Delta K}\left(V^n_{i,j} - V^n_{i,j-1}\right) - \frac{b^-_{i,j}}{\Delta K}\left(V^n_{i,j+1} - V^n_{i,j}\right)\end{aligned}$$

where

$$\begin{aligned} a_{i,j} &= -\frac{1-\omega}{\omega}(V_k)^{1/\omega-1}_{i,j} - (1-\tau)(\theta\lambda K^{\theta-1}_{i,j} - \delta)k_{i,j} - \lambda K^\theta_{i,j}(1-\theta) \\ b_{i,j} &= -\lambda K^\theta + \delta K + V_k(K,K)^{1/\omega-1} + \tau(\theta\lambda K^\theta - \delta K)\end{aligned}$$

Results Using 2-D Upwind Method: τ Variation

- $\theta = 0.5$, $\omega = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$
- Aggregate consumption function for various values of τ:



Results Using 2-D Upwind Method: τ Variation

- $\theta = 0.5$, $\omega = 0.5$, $\rho = 0.05$, $\lambda = 1.0$, $\delta = 0.025$, $\sigma = 0.0$
- For $\tau = 0.0$, compare to exact solution: