Chemical Solvers
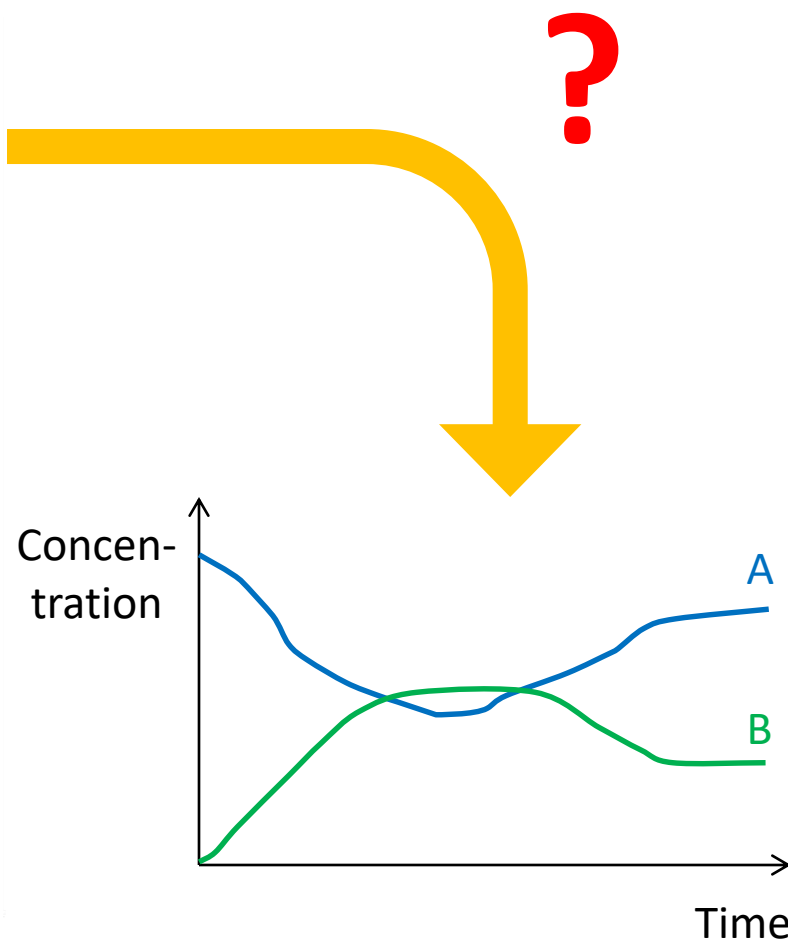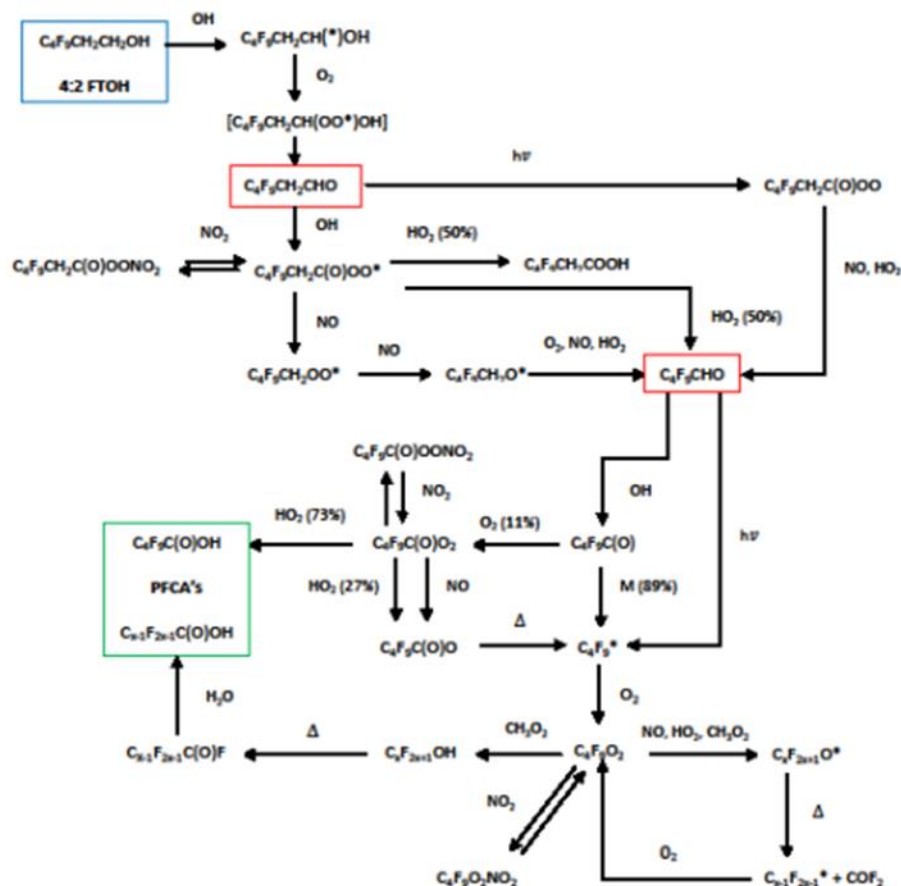
Oliver Wild

Lancaster Environment Centre

How can we model chemical processes?

Chemical Reaction Kinetics

$$aA + bB \longrightarrow cC + dD$$

$$\text{rate} = k\,[A]^a\,[B]^b$$

Reaction rate

Rate constant

Concentration of reactants

$$\frac{d[C]}{dt} = \frac{-d[A]}{dt}\,\frac{c}{a}$$

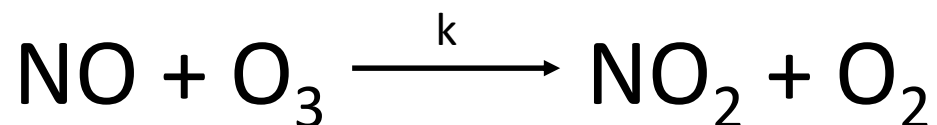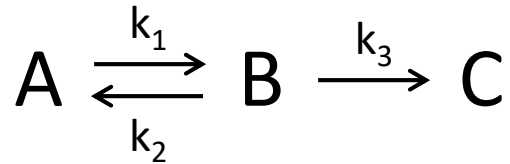$$k = Ae^{-E_a/(RT)}$$

Chemical Reaction Kinetics

$$aA + bB \longrightarrow cC + dD$$

$$\text{rate} = k\,[A]^a\,[B]^b$$

Example:

$$NO + O_3 \xrightarrow{\quad k \quad} NO_2 + O_2$$

$$\frac{d[NO]}{dt} = -k\,[NO]\,[O_3]$$

Differential equation …

Ordinary Differential Equations

$$A \underset{k_2}{\overset{k_1}{\rightleftharpoons}} B \xrightarrow{k_3} C$$

$d[A]/dt = -k_1[A] + k_2[B]$

$d[B]/dt = k_1[A] - (k_2 + k_3)[B]$

$d[C]/dt = k_3[B]$

time
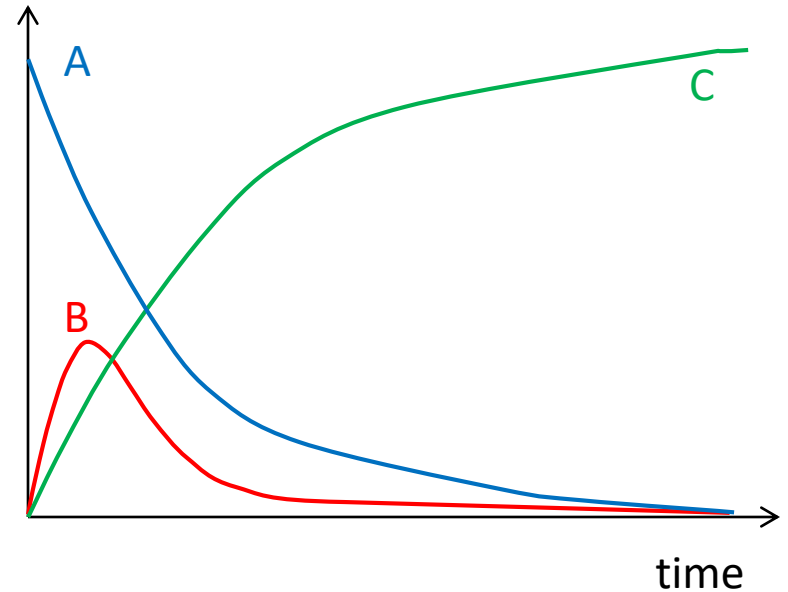
- Chemical kinetics leads to a system of coupled ODEs
- These need to be solved simultaneously

Solving ODEs by Numerical Integration

$$A \xrightarrow{k_1} B$$
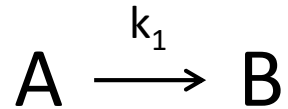
$$d[A]/dt = -k_1[A]$$

$$\int_{A_0}^{A_t} d[A]/[A] = -k_1 \int_{t_0}^{t} dt$$

$$[\ln(A)]_{A_0}^{A_t} = -k_1[t]_{t_0}^{t}$$

$$\ln(A_t/A_0) = -k_1(t-t_0)$$

$$A_t = A_0\, e^{-kt} \quad \text{(for } t_0=0\text{)}$$

- First-order loss gives simple exponential decay
- Full chemistry ODEs generally can't be solved analytically

Euler Method

- Simple and easy to apply
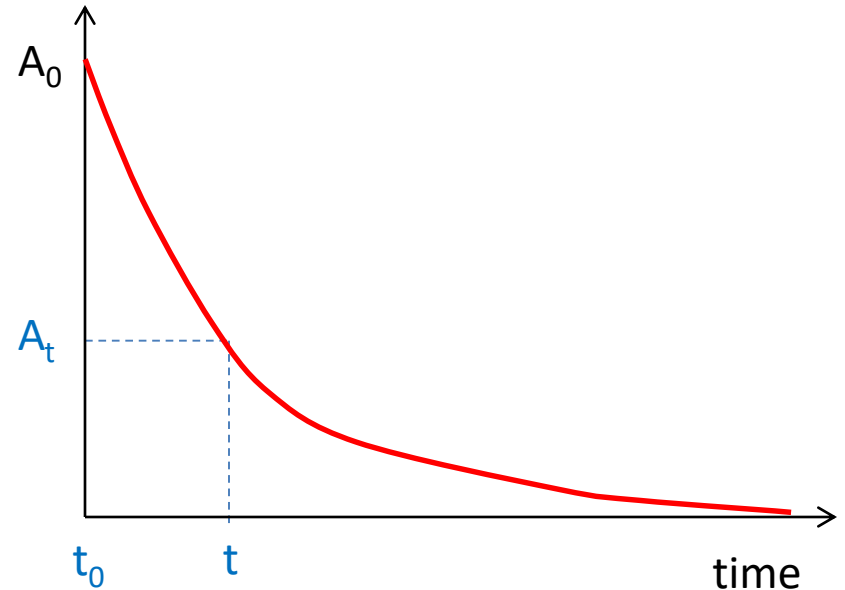 - Evaluate gradient at current state
  - Move forward in time
  - Re-evaluate gradient at new state
  - Repeat...

- Small time steps for accuracy

- Problems with instability
  - May get oscillations
  - Doesn't always converge

- Improvements
  - Midpoint method
  - Backward Euler (Implicit)

$$d[A]/dt = -k_1[A]$$

$$\Delta A = -k_1 A \Delta t$$

$$dA/dt = -2.3A$$
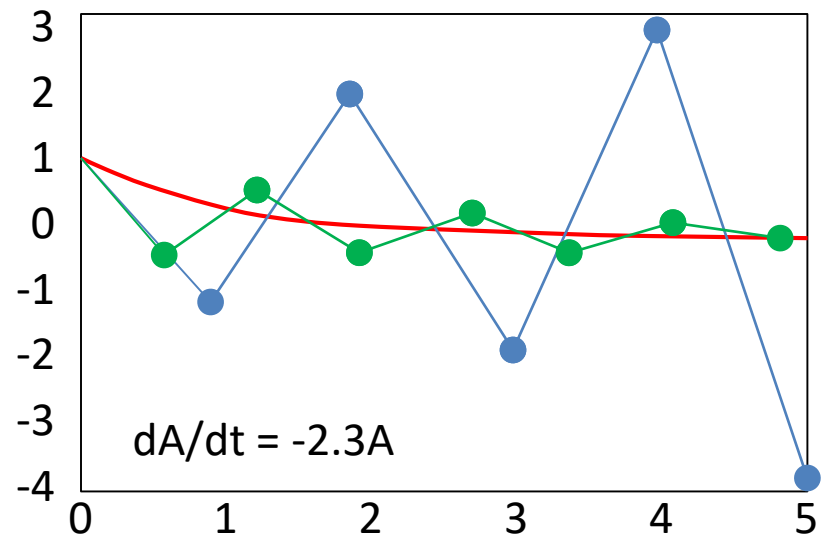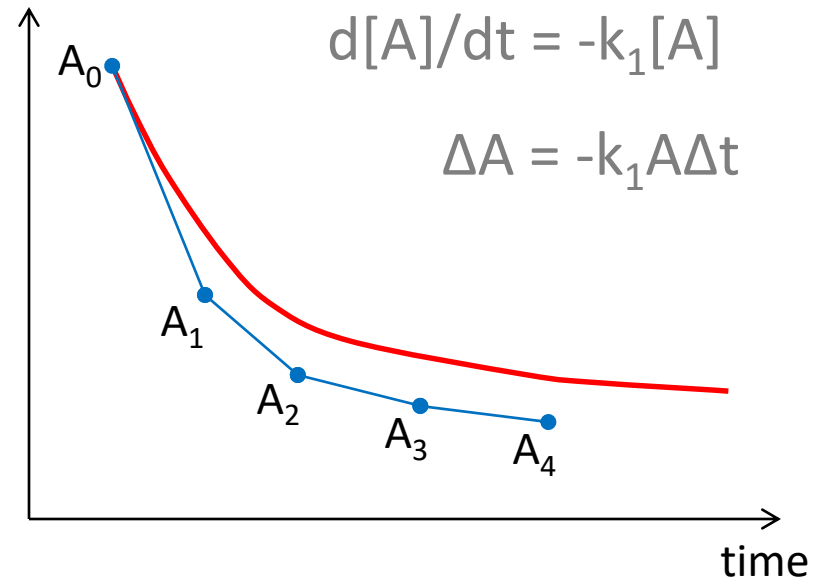
Explicit vs. Implicit Methods

- **Explicit methods**
  - Calculate state of system at later time from state at current time
  - A(t+Δt) = F(A(t))
  - Advantage: easy to implement
  - Disadvantage: need very small Δt if equations are stiff

- **Implicit methods**
  - Use both current state of system and later one
  - G(A(t),A(t+Δt)) = 0
  - Disadvantage: extra calculation required to solve eqn.
  - Advantage: can use much larger time steps; more stable

Best approach to use depends on the problem to be solved

Stiffness important; often a trade-off of accuracy vs. stability

Backward Euler Method

- Use gradient at final state

  $A_{n+1} = A_n + f(t_{n+1}, A_{n+1})\Delta t$

  e.g., $A_1 = A_0 - kA_1\Delta t$

- A_1 appears on both sides, so need to solve equation

- Typically set $A_1 = A_0$ and then re-evaluate until it converges
  - Newton-Raphson iteration

- Stable approach
  - Good for stiff systems
  - Can take large steps

$d[A]/dt = -k_1[A]$

A_0

A_1

time
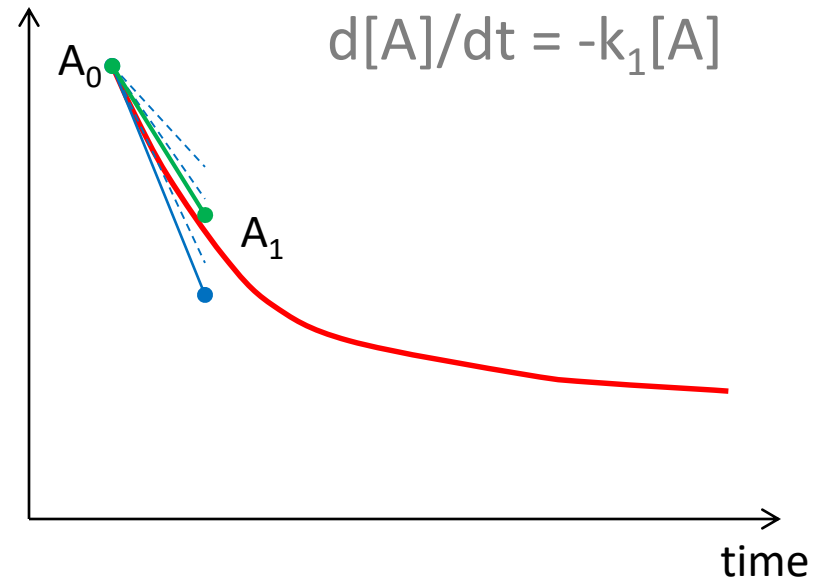
Find A_{n+1} such that

$A_{n+1} - A_n - f(t_{n+1}, A_{n+1})\,\Delta t = 0$

Newton-Raphson Iteration
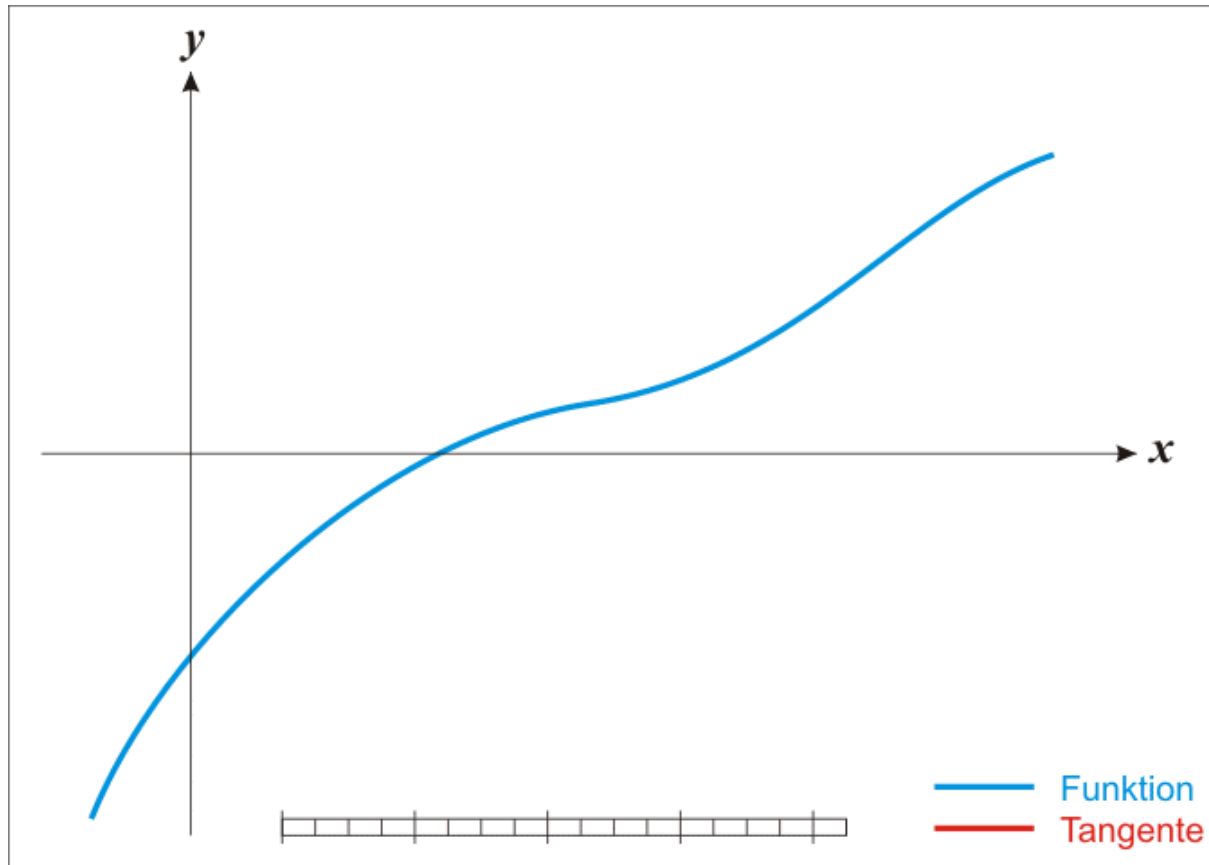
- Find root of function (y=0): take first guess, calculate gradient, find x-axis crossing point, then repeat iteratively...

   (need a good first guess or may fail to converge in some cases)

Other Integration Methods

- Adams-Bashforth
  - Explicit, linear multi-step methods
  - $A = A_0 - kA_0\Delta t$ $A_1 = A - (1.5kA - 0.5kA_0)\Delta t$

- Runge-Kutta methods
  - Iterative, implicit or explicit methods
  - $A_1 = A_0 + (y_1 + 2y_2 + 2y_3 + y_4) * \Delta t/6$

- Gear methods (backward differentiation formulae)
  - Implicit, linear multi-step methods
  - Commonly used in, e.g., SVODE, smvgear

- Rosenbrock solvers
  - Multi-step methods, similar to Runge-Kutta
  - e.g., RODAS

Read a good book on Numerical Methods to learn more!

Stiffness

If a numerical method with a finite region of absolute stability, applied to a system with any initial conditions, is forced to use in a certain interval of integration a step length which is excessively small in relation to the smoothness of the exact solution in that interval, then the system is said to be stiff in that interval.
   J.D. Lambert

Equation systems may be stiff when
  – some species decay much more rapidly than others
  – the step length is constrained by stability rather than accuracy
  – there is strong coupling between different species

We can often alter the stiffness of a system by putting species in steady state or in chemical families

Steady State and Chemical Families

- ## Steady State

$$A \xrightarrow{k_1} B \xrightarrow{k_2} C \qquad \text{if } k_2 \gg k_1 \quad \text{B removed as soon as formed}$$
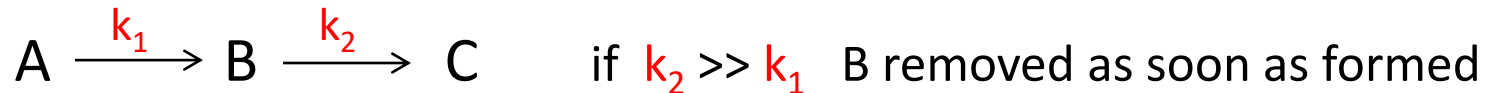
If no other reactions, B quickly reaches equilibrium (steady state)

➢ $d[B]/dt = k_1[A] - k_2[B] = 0$ so $[B] = (k_1/k_2)[A]$

➢ $d[C]/dt = k_2[B] = k_1[A]$ as k_1 is the rate limiting step

Don't need to solve for [B] if k_2 very fast – just function of [A]

- ## Chemical Families

$$O_2 \xrightarrow{k_1} O \underset{k_3}{\overset{k_2}{\rightleftarrows}} O_3 \xrightarrow{k_4} O_2 \qquad \text{if } k_2, k_3 \gg k_1, k_4$$

O_x
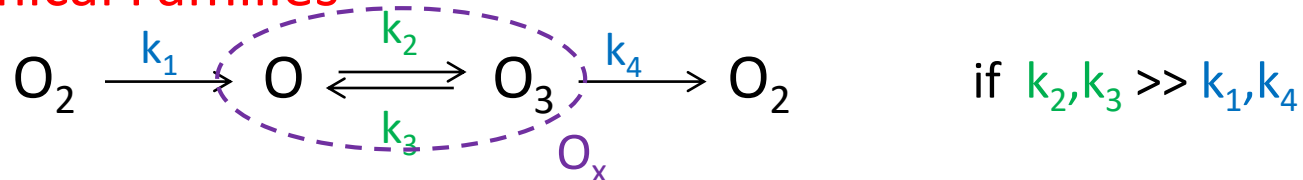
Interconversion of O and O_3 very fast – treat as a single family, O_x

➢ solve for O_x based on k_1 and k_4

➢ derive O and O_3 from O_x based on ratio of k_2 and k_3

What should we include in our chemistry?
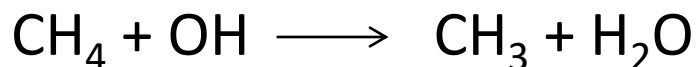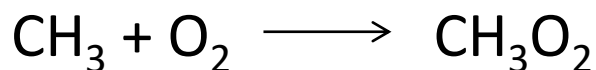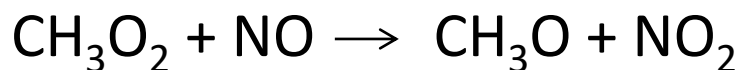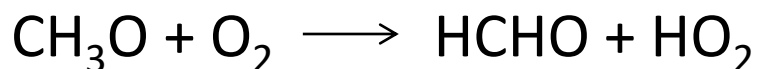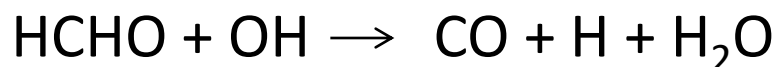
Atmospheric oxidation of CH_4 species lifetime

$$CH_4 + OH \longrightarrow CH_3 + H_2O$$ $\tau \approx 10$ yrs

$$CH_3 + O_2 \longrightarrow CH_3O_2$$ $\tau \approx 1$ ms

$$CH_3O_2 + NO \rightarrow CH_3O + NO_2$$ $\tau \approx 100$ s

$$CH_3O + O_2 \longrightarrow HCHO + HO_2$$ $\tau \approx 1$ s

$$HCHO + OH \rightarrow CO + H + H_2O$$ $\tau \approx 1$ day

$$HCHO + h\nu \rightarrow CO + 2H$$

$$CO + OH \longrightarrow CO_2 + H$$ $\tau \approx 3$ months

$$H + O_2 \longrightarrow HO_2$$ $\tau \approx 1$ ms

What should we include in our chemistry?

Atmospheric oxidation of CH_4 species lifetime

$$CH_4 + OH \longrightarrow CH_3 + H_2O \qquad \tau \approx 10 \text{ yrs}$$

$$CH_3 + O_2 \longrightarrow CH_3O_2 \qquad \tau \approx 1 \text{ ms}$$

$$CH_3O_2 + NO \longrightarrow CH_3O + NO_2 \qquad \tau \approx 100 \text{ s}$$

$$CH_3O + O_2 \longrightarrow HCHO + HO_2 \qquad \tau \approx 1 \text{ s}$$

$$HCHO + OH \longrightarrow CO + H + H_2O \qquad \tau \approx 1 \text{ day}$$

$$HCHO + h\nu \longrightarrow CO + 2H$$

$$CO + OH \longrightarrow CO_2 + H \qquad \tau \approx 3 \text{ months}$$
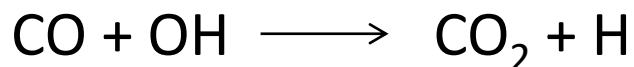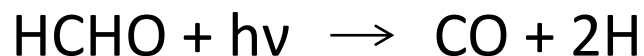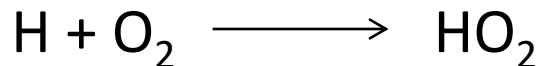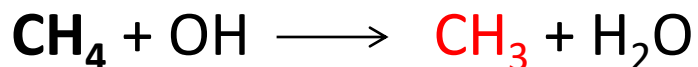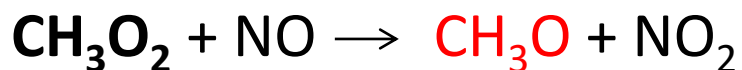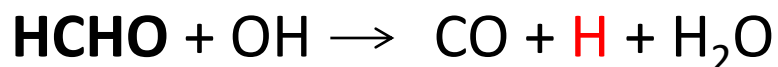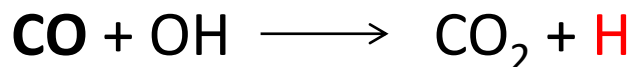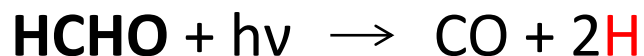
$$H + O_2 \longrightarrow HO_2 \qquad \tau \approx 1 \text{ ms}$$

Drop the very fast reactions and very short-lived species, replacing the latter with their ultimate oxidation products (e.g., HO_2 for H)

ASAD Chemistry Package

- Framework for including chemistry schemes in models
    - Contains choice of numerical integration package
    - Independent of chemistry scheme
    - Solve simultaneous differential equation for each species, *y*:

$$\frac{dy}{dt} = P - Ly - ly^2 + E - Dy$$

    - Typically treat emissions (E) and deposition (D) outside ASAD

ASAD Framework

- Set up chemistry
  - Read species and reaction variables
  - Initialise arrays for production and loss terms

```
TYPE(CHCH_T), PUBLIC :: chch_defs_strattrop_aer(1:87)=(/          &
!   1
chch_t(  1,'O(3P)       ',  1,'TR          ','Ox          ',  0,  0,  0),  &
!   2
chch_t(  2,'O(1D)       ',  1,'SS          ','Ox          ',  0,  0,  0),  &
!   3 DD: 1,WD: 1,
chch_t(  3,'O3          ',  1,'TR          ','Ox          ',  1,  1,  0),  &
!   4
chch_t(  4,'N           ',  1,'TR          ','NOx         ',  0,  0,  0),  &
!   5 DD: 2,
chch_t(  5,'NO          ',  1,'TR          ','NOx         ',  1,  0,  0),  &
!   6 DD: 3,WD: 2,
chch_t(  6,'NO3         ',  1,'TR          ','NOx         ',  1,  1,  0),  &
!   7 DD: 4,      EM: 1
chch_t(  7,'NO2         ',  1,'TR          ','NOx         ',  1,  0,  1),  &

!  74
chch_t( 74,'O2          ',  1,'CT          ','            ',  0,  0,  0),  &
!  75
chch_t( 75,'N2          ',  1,'CT          ','            ',  0,  0,  0),  &
```

TR = tracer
SS = steady state
CT = constant

Bimolecular Reactions

$$k = Ae^{-E_a/(RT)}$$

$$k = k_0 \left(\frac{T}{300}\right)^{\alpha} \exp\left(\frac{-\beta}{T}\right)$$

Arrhenius equation

```
ratb_t('Reactant 1','Reactant 2','Product 1 ','Product 2 ','Product 3 ',&
'Product 4 ',  k₀,  α,  β, Fraction of Product 1 produced, Fraction of Product 2 produced, &
Fraction of Product 3 produced, Fraction of Product 4 produced), &
```

```
ratb_t('O3        ','C5H8      ','HO2        ','OH        ','         ',& ! B133
'         ', 3.33E-15,  0.00,   1995.00, 0.750, 0.750, 0.000, 0.000), & ! B133 IUPAC2007*
...
ratb_t('OH        ','C5H8      ','ISO2      ','        ','         ',& ! B144
'         ', 2.70E-11,  0.00,   -390.00, 0.000, 0.000, 0.000, 0.000), & ! B144 IUPAC2009
...
```

Termolecular Reactions

(and unimolecular decomposition)

$$k = \left(\frac{k_0\,[M]}{1 + k_0\,[M]\,/\,k_\infty} \right) F_c^{\left(1 + \left[\log_{10}\left(\frac{k_0[M]}{k_\infty} \right) \right]^2 \right)^{-1}}$$

$$k_0 = k_1 \left(\frac{T}{300} \right)^{\alpha_1} \exp\left(\frac{-\beta_1}{T} \right)$$

$$k_\infty = k_2 \left(\frac{T}{300} \right)^{\alpha_2} \exp\left(\frac{-\beta_2}{T} \right)$$

$$F_c = \exp\left(-T/f \right) \quad or \quad F_c = f$$

```
ratt_t('Reactant 1','Reactant 2','Product 1 ','Product 2 ', f, &
k₁,  α₁,  β₁, k₂,  α₂,  β₂, Fraction of Product 1 produced, Fraction of Product 2 produced), &
```

```
ratt_t('N2O5       ','m          ','NO2        ','NO3        ',      0.3,    & ! T023
  1.30E-03, -3.50, 11000.00,  9.70E+14,  0.10, 11080.00, 0.000, 0.000), & ! T023 IUPAC 2002
ratt_t('NO         ','NO         ','NO2        ','NO2        ',      0.0,    & ! T024
  3.30E-39,  0.00,  -530.00,  0.00E+00,  0.00,     0.00, 0.000, 0.000)  & ! T024 IUPAC 2001
```

Photolysis

- Photodissociation processes important for many species

- Not part of ASAD, calculated using Fast-Jx based on
  - Absorption x-section
  - Actinic Flux, hv

accounting for:

  - Solar zenith angle
  - Wavelength dependence
  - Scattering and absorption by molecules, aerosol, clouds
  - Surface albedo
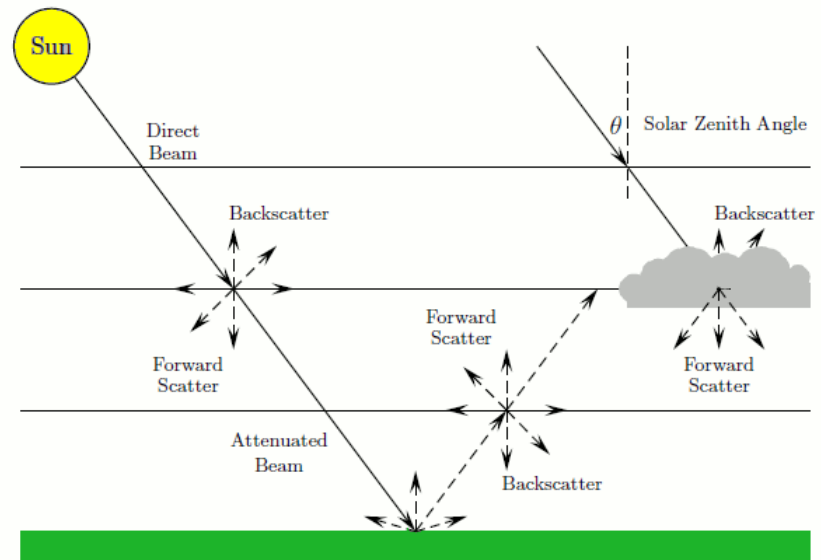
ASAD Framework

- Set up chemistry
  - Read species and reaction variables
  - Initialise arrays for production and loss terms

- Loop over chemical time step
  - Collect concentrations (molecules/cm^3)
  - Calculate rate constants (based on temperature, etc.)
  - Integrate chemistry ODEs
    - Calculate production and loss rates
    - Integrate with chosen integration scheme
  - Determine new concentrations

Chemical solver

Coding ODEs

```fortran
C
C
C     FIRST WORK OUT O3EQUIL AND TIME CONSTANT
C
      FCL=(1.+RHCL(I,J)+RCLOCL(I,J)+RCNCL(I,J)+RHOCL(I,J)+
     1      ROCLO(I,J)+2.0*RCL2O2(I,J))
      FBR=(1. + RHBR(I,J) + RBROBR(I,J) + RBNBR(I,J) + RHOBR(I,J))
      BR=BROX(I,J)/FBR
      CL=CLOX(I,J)/FCL
      AA=2.*K3T*RO03(I,J)*DLITE(I)
      CC=DLITE(I)*(-2.*J2(I,J)*O2-JNO2(I,J)*NO2(I,J)
     1      -K14T*OHSTOR**2 - ALFA*K20*HNO3(I,J)*OHSTOR
     2      -((JNO(I,J)+K70T*RNNO2(I,J)*NO2(I,J))*RNONO2(I,J)+
     3       K71T*RNNO2(I,J)*O2)*NO2(I,J)
     3      -ALFA*JN2O5(I,J)*N2O5(I,J)-ALFA*K92T(I,J)*N2O5(I,J)
     4      -ALFA*JCNIT(I,J)*RCNCL(I,J)*CL
     5      -JBRO(I,J)*RBROBR(I,J)*BR-JOCLO(I,J)*ROCLO(I,J)*CL)
   50 BB=DLITE(I)*((K12*RHOH(I,J)+K11T+K36T*RHO2OH(I,J)
     1      +(K8+K10*RHO2OH(I,J))
     1      *RO03(I,J))*OHSTOR + (K17T*RNONO2(I,J) + K16*RO03(I,J))
     2      + (1.-ALFA)*K18T + K72T*RNNO2(I,J))*NO2(I,J)
     3      +(K7*H2O(I,J)+K31*CH4(I,J)+K32*H2+(K22+K22B)*N2O(I,J))
     4      +K48*F11(I,J)+K49*F12(I,J))*RO1DO3(I,J)
     5      +(K50T+K51T*RO03(I,J)*RCLOCL(I,J)+(1.-ALFA)*K63T*
     6      RO03(I,J)*RCNCL(I,J)+K67T*RO03(I,J)*RHOCL(I,J))*CL
     7      +K107*RC2OCH(I,J)*CH4(I,J)*RO03(I,J)
     7      +K111*CH4(I,J)*RO1DO3(I,J)
     7  +K120*BR+K121*RO03(I,J)*RBROBR(I,J)*BR+K164*C2H4(I,J))
      B=(1.-DLITE(I))*(K18T*NO2(I,J)*(1.+RNONO2(I,J)))
      BBB=BB+B
      ARG=BBB*BBB-4.*AA*CC
      IF(ARG.GT.0.0) GO TO 55
      IF(KOUNT.LT.300) WRITE(6,1724) NSTEP,I,J,AA,BB,CC
      KOUNT=KOUNT + 1
      ARG=BBB*BBB
   55 O3DRAT=-(CC+BB*O3STOR+AA*O3STOR**2)
      O3NRAT=-(B*O3STOR)
      IF (DLITE(I).LT.0.0000001) GOTO 980
      O3(I,J)=(-BBB+SQRT(ARG))/(2.*AA)
      IF((ABS((O3(I,J)-O3STOR)/(O3DRAT+O3NRAT))
     1    .LT. 2.88E4) )                        GO TO 60
C EXPRESSION IN PARENTHESIS IS O3 EQUILIBRIUM TIME CONSTANT
C O3 DEEMED TO BE IN EQUILIBRIUM IF TIME CONSTANT<8 HOURS(2.88E4 SECS)
  980    O3RT(I,J)=(O3DRAT+O3NRAT)*COFO
      GO TO 70
   60 O3UM(I,J)=O3(I,J)/M(I,J)
      LEV=MIN0(LEV,J)
      IF(J.GT.4) GO TO 70
      O3UM(I,J)=1.0
      O3RT(I,J)=(O3DRAT+O3NRAT)*COFO
   70 CONTINUE
C
C NOW WORK OUT THE REST OF THE RATES
      NO2RT(I,J)=(2.*K22*N2O(I,J)*RO1DO3(I,J)*O3STOR+(JHNO3(I,J)
     1      +K20*OHSTOR)*HNO3(I,J)-K21(I,J)*NO2(I,J)*OHSTOR
     2      -K37T(I,J)*OHSTOR*RHO2OH(I,J)*NO2(I,J)+(JPNA(I,J)
     3      +K39T(I,J)+K40T*OHSTOR)*PNA(I,J)
     4      -2.*K70T*RNNO2(I,J)*RNONO2(I,J)*NO2(I,J)**2
```

Production and loss rates

- All species hard-wired

- All reactions hard-wired

- Needs to be hand-written
- Not very flexible!

Two innovations

- Machine-written code
  - Simpler to make changes
- Symbolic arrays
  - Increased flexibility

```fortran
Cwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwww
C
C
C        FIRST WORK OUT O3EQUIL AND TIME CONSTANT
C
      FCL=(1.+RHCL(I,J)+RCLOCL(I,J)+RCNCL(I,J)+RHOCL(I,J)+
     1        ROCLO(I,J)+2.0*RCL2O2(I,J))
      FBR=(1. + RHBR(I,J) + RBROBR(I,J) + RBNBR(I,J) + RHOBR(I,J))
      BR=BROX(I,J)/FBR
      CL=CLOX(I,J)/FCL
      AA=2.*K3T*ROO3(I,J)*DLITE(I)
      CC=DLITE(I)*(-2.*J2(I,J)*O2-JNO2(I,J)*NO2(I,J)
     1     -K14T*OHSTOR**2 - ALFA*K20*HNO3(I,J)*OHSTOR
     2     -((JNO(I,J)+K70T*RNNO2(I,J)*NO2(I,J))*RNONO2(I,J)+
     3      K71T*RNNO2(I,J)*O2)*NO2(I,J)
     3     -ALFA*JN2O5(I,J)*N2O5(I,J)-ALFA*K92T(I,J)*N2O5(I,J)
     4     -ALFA*JCNIT(I,J)*RCNCL(I,J)*CL
     5     -JBRO(I,J)*RBROBR(I,J)*BR-JOCLO(I,J)*ROCLO(I,J)*CL)
   50 BB=DLITE(I)*((K12*RHOH(I,J)+K11T+K36T*RHO2OH(I,J)
     1     +(K8+K10*RHO2OH(I,J))
     1     *ROO3(I,J))*OHSTOR + (K17T*RNONO2(I,J) + K16*ROO3(I,J)
     2     + (1.-ALFA)*K18T + K72T*RNNO2(I,J))*NO2(I,J)
     3     +(K7*H2O(I,J)+K31*CH4(I,J)+K32*H2+(K22+K22B)*N2O(I,J)
     4     +K48*F11(I,J)+K49*F12(I,J))*RO1D3(I,J)
     5     +(K50T+K51T*ROO3(I,J)*RCLOCL(I,J)+(1.-ALFA)*K63T*
     6      ROO3(I,J)*RCNCL(I,J)+K67T*ROO3(I,J)*RHOCL(I,J))*CL
     7     +K107*RC2OCH(I,J)*CH4(I,J)*ROO3(I,J)
     7     +K111*CH4(I,J)*RO1D3(I,J)
     7   +K120*BR+K121*ROO3(I,J)*RBROBR(I,J)*BR+K164*C2H4(I,J))
      B=(1.-DLITE(I))*(K18T*NO2(I,J)*(1.+RNONO2(I,J)))
      BBB=BB+B
      ARG=BBB*BBB-4.*AA*CC
      IF(ARG.GT.0.0) GO TO 55
      IF(KOUNT.LT.300) WRITE(6,1724) NSTEP,I,J,AA,BB,CC
      KOUNT=KOUNT + 1
      ARG=BBB*BBB
   55 O3DRAT=-(CC+BB*O3STOR+AA*O3STOR**2)
      O3NRAT=-(B*O3STOR)
      IF (DLITE(I).LT.0.0000001) GOTO 980
      O3(I,J)=(-BBB+SQRT(ARG))/(2.*AA)
      IF((ABS((O3(I,J)-O3STOR)/(O3DRAT+O3NRAT))
     1     .LT. 2.88E4) )                       GO TO 60
C EXPRESSION IN PARENTHESIS IS O3 EQUILIBRIUM TIME CONSTANT
C O3 DEEMED TO BE IN EQUILIBRIUM IF TIME CONSTANT<8 HOURS(2.88E4 S
  980    O3RT(I,J)=(O3DRAT+O3NRAT)*COFO
      GO TO 70
   60 O3UM(I,J)=O3(I,J)/M(I,J)
      LEV=MINO(LEV,J)
      IF(J.GT.4) GO TO 70
      O3UM(I,J)=1.0
      O3RT(I,J)=(O3DRAT+O3NRAT)*COFO
   70 CONTINUE
C
C NOW WORK OUT THE REST OF THE RATES
      NO2RT(I,J)=(2.*K22*N2O(I,J)*RO1D3(I,J)*O3STOR+(JHNO3(I,J)
     1        +K20*OHSTOR)*HNO3(I,J)-K21*O(I,J)*NO2(I,J)*OHSTOR
     2        -K37T(I,J)*OHSTOR*RHO2OH(I,J)*NO2(I,J)+(JPNA(I,J)
     3        +K39T(I,J)+K40T*OHSTOR)*PNA(I,J)
     4        -2.*K70T*RNNO2(I,J)*RNONO2(I,J)*NO2(I,J)**2
```

```fortran
c------------------------------------------------
c  bimolecular reactions
c
      loss(jl,   1)=loss(jl,   1)-kb(jl,38  )*y(jl,25 )*m(jl)-kb(jl,
     *       39  )*y(jl,25 )*m(jl)-kb(jl,40  )*y(jl,8  )*m(jl)-kb(j
     *       l,41  )*y(jl,7  )*m(jl)-kb(jl,42  )*y(jl,12 )*m(jl)-kb
     *       (jl,43  )*y(jl,6  )*m(jl)-kb(jl,44  )*y(jl,21 )*m(jl)-
     *       kb(jl,45  )*y(jl,20 )*m(jl)-kb(jl,46  )*y(jl,33 )*m(jl
     *       )-kb(jl,47  )*y(jl,19 )*m(jl)-kb(jl,48  )*y(jl,17 )*m(
     *       jl)-kb(jl,49  )*y(jl,18 )*m(jl)-kb(jl,50  )*y(jl,2  )*
     *       m(jl)-kb(jl,51  )*y(jl,5  )*m(jl)
      prod(jl,   1)=prod(jl,   1)+kb(jl,8  )*y(jl,4  )*y(jl,6  )*m(
     *       jl)+kb(jl,55  )*y(jl,3  )*y(jl,24 )*m(jl)+kb(jl,56  )*
     *       y(jl,3  )*y(jl,23 )*m(jl)+kb(jl,57  )*y(jl,2  )*y(jl,3
     *       )*m(jl)+kb(jl,57  )*y(jl,2  )*y(jl,3  )*m(jl)+kb(jl,
     *       71  )*y(jl,5  )*y(jl,5  )*m(jl)+kb(jl,127 )*y(jl,23 )*
     *       y(jl,61 )*m(jl)
      loss(jl,   2)=loss(jl,   2)-kb(jl,2  )*y(jl,8  )*m(jl)-kb(jl,
     *       10  )*y(jl,4  )*m(jl)-kb(jl,18  )*y(jl,6  )*m(jl)-kb(j
     *       l,27  )*y(jl,14 )*m(jl)-kb(jl,30  )*y(jl,16 )*m(jl)-kb
     *       (jl,32  )*y(jl,17 )*m(jl)-kb(jl,50  )*y(jl,1  )*m(jl)-
     *       kb(jl,57  )*y(jl,3  )*m(jl)-kb(jl,58  )*y(jl,3  )*m(jl
     *       )-kb(jl,59  )*y(jl,25 )*m(jl)-kb(jl,70  )*y(jl,5  )*m(
     *       jl)-kb(jl,87  )*y(jl,26 )*m(jl)-kb(jl,89  )*y(jl,27 )*
     *       m(jl)-kb(jl,115 )*y(jl,53 )*m(jl)-kb(jl,119 )*y(jl,55
     *       )*m(jl)-kb(jl,123 )*y(jl,57 )*m(jl)-kb(jl,128 )*y(jl,6
     *       1 )*m(jl)-kb(jl,131 )*y(jl,62 )*m(jl)-kb(jl,133 )*y(jl
     *       ,66  )*m(jl)-kb(jl,142 )*y(jl,69 )*m(jl)-kb(jl,144 )*y(
     *       jl,71 )*m(jl)-kb(jl,146 )*y(jl,72 )*m(jl)
      prod(jl,   2)=prod(jl,   2)
      loss(jl,   3)=loss(jl,   3)-kb(jl,52  )*y(jl,9  )*m(jl)-kb(jl,
     *       53  )*y(jl,9  )*m(jl)-kb(jl,54  )*y(jl,22 )*m(jl)-kb(j
     *       l,55  )*y(jl,24 )*m(jl)-kb(jl,56  )*y(jl,23 )*m(jl)-kb
     *       (jl,57  )*y(jl,2  )*m(jl)-kb(jl,58  )*y(jl,2  )*m(jl)
      prod(jl,   3)=prod(jl,   3)
```

```fortran
c  Calculate Bimolecular reaction rates
      do jr = 1, jpnr
        do jl = jl1, jl2
          prk(jl,jr) = rk(jl,jr) * y(jl,nspi(jr,1)) * y(jl,nspi(jr,2))
        enddo
      enddo
c
c Calculate production and loss rate for each species
      do j = 1, knspec
        do jl = jl1, jl2
          pd(jl,kspec(j)) = pd(jl,kspec(j)) + prk(jl,kspec(j))
        enddo
      enddo
c
```

Backward Euler Scheme with N-R Iteration

- **Build Jacobian**

$$J_{ij} = \frac{\partial F_i}{\partial x_j}$$

  - Matrix of first-order partial derivatives
  - This defines system of coupled nonlinear equations

- **Solve** $\quad J_F(x_n)(x_{n+1} - x_n) = -F(x_n)$

  - Solve for $(x_{n+1} - x_n)$ This is our A_{n+1}-A_n term from earlier

- **Use Gaussian Elimination**   (LU factorization with partial pivoting)
  - Forward elimination: rearrange rows to form a triangular matrix
  - Back substitution: continue row operations to solve
  - Speed this up using a sparse matrix approach
    - Only operate on non-zero elements to reduce computational demands

Backward Euler Scheme with N-R Iteration

- Build Jacobian
- Make first guess at new concentrations: x_1 (use Euler method)
- Iteration loop
  - Update rates of change: dx/dt
  - Calculate residual term to minimise: $(x_1 - x_0) - dx/dt.\Delta t$
  - Check for convergence within specified tolerances
  - Rebuild Jacobian
  - Solve by Gaussian Elimination: $(x_1 - x_0)$
  - Check for any problems (slow convergence, divergence, etc.)
  - Update concentrations: x_1
- If no convergence, retry with reduced time step
- Otherwise, conclude successfully

Integration Scheme Features

- Solution time roughly linear in number of entries in Jacobian
  - Previously proportional to square of number of species
  - Adding a few species or reactions shouldn't affect time much…
    … unless new species coupling increases number of iterations needed

- Should converge in 5-7 iterations in most cases
  - Initial iterations damped to accelerate convergence (?)
  - Maximum number of iterations limited (drop out if too many!)
  - Halving time step improves convergence (load balancing issues?)
  - Can alter tolerances, but do so with care!

- Efficiency improved by working on multiple grid cells at once
  - Optimized for vector processing

What do I need to know to run UKCA?

- In most cases: almost nothing!

- Requirements for new chemistry in ASAD
    - How to add new species and new reactions   (see tutorial)

- The NR chemical solver is very robust, but note:
    - Increased stiffness (v. short-lived species) may give convergence issues
        - Consider if short-lived species needed, or if they should be in steady state
    - New steady state species may require additional work
        - Rates need to be included in Jacobian (tweaking may be necessary!)

- If you're interested in numerical methods, you're welcome to dig a little deeper into these parts of UKCA…