

Prozessdynamik  
Hausaufgabe zur 5. Übung

Steffi Klinge  
STK 03  
XXXXXX

18. Januar 2005

1.) Die allgemeine partielle Massenbilanz für die Komponente  $\alpha$  lautet

$$\frac{dm_\alpha}{dt} = M_{in,\alpha} - M_{out,\alpha} + V_R \sigma_\alpha^m$$

In unserem Fall ergibt sich wegen  $\bar{M} = \bar{M}_A = \bar{M}_B = const.$  die allgemeine partielle Stoffmengenbilanz nach der Division durch  $\bar{M}$  zu

$$\frac{dn_\alpha}{dt} = G_{in,\alpha} - G_{out,\alpha} + V_R \sigma_\alpha$$

Es gilt weiterhin  $G_\alpha = F \cdot c_\alpha$  und  $n_\alpha = V_R \cdot c_\alpha$  und somit

$$\frac{d(V_R c_\alpha)}{dt} = F_{in} c_\alpha^{in} - F_{out} c_\alpha + V_R \sigma_\alpha$$

Im betrachteten Beispiel gibt es zwei Zuläufe —  $F_{in,1}$  und  $F_{in,2}$  — und einen Ablauf  $F_{out}$ . Da das Reaktorvolumen  $V_R$  konstant ist, muss hier gelten  $F_{out} = F_{in,1} + F_{in,2}$ . Es folgt daraus die dynamische Gleichung für die Konzentration von  $\alpha$

$$\frac{dc_\alpha}{dt} = \frac{1}{V_R} (F_{in,1} (c_\alpha^{in,1} - c_\alpha) + F_{in,2} (c_\alpha^{in,2} - c_\alpha)) + \sigma_\alpha$$

Bei der vorliegenden Reaktion  $2A \rightleftharpoons 2B$  erhält man mit  $\sigma_\alpha = \sum_j \nu_{\alpha,j} r_j$  und  $r_1 r_{1+} - r_{1-} = k_{1+} \cdot c_A^2 - k_{1-} \cdot c_B^2$  die dynamischen Gleichungen für die Konzentrationen von  $A$  und  $B$

$$\frac{dc_A}{dt} = \frac{1}{V_R} (F_{in,1} (c_A^{in,1} - c_A) + F_{in,2} (c_A^{in,2} - c_A)) - 2k_{1+} \cdot c_A^2 + 2k_{1-} \cdot c_B^2$$

$$\frac{dc_B}{dt} = \frac{1}{V_R} (-F_{in,1} c_B + F_{in,2} (c_B^{in,2} - c_B)) + 2k_{1+} \cdot c_A^2 - 2k_{1-} \cdot c_B^2$$

2.) Die allgemeine Form des Newton-Verfahrens lautet

$$x^{(k+1)} = x^{(k)} - (J^{(k)})^{-1} \cdot f(x^{(k)})$$

Um dieses Verfahren zur Berechnung des stationären Zustandes ( $f = \frac{dc}{dt} = 0$ ) anzuwenden, muss die Jacobi-Matrix  $J$  mit den Elementen  $J_{ij} = \frac{\partial f_i}{\partial x_j}$  ermittelt werden.

$$J = \begin{bmatrix} -\frac{F_{in,1}+F_{in,2}}{V_R} - 4k_{1+} \cdot c_A & 4k_{1-} \cdot c_B \\ 4k_{1+} \cdot c_A & -\frac{F_{in,1}+F_{in,2}}{V_R} - 4k_{1-} \cdot c_B \end{bmatrix}$$

Die Inverse einer  $2 \times 2$  - Matrix  $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$  wird berechnet mittels  $A^{-1} = \frac{1}{ad-bc} \cdot \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$ .

Somit erhält man die benötigte Inverse Jacobi-Matrix und es ergibt sich die Iterationsvorschrift

$$\begin{bmatrix} c_A^{(k+1)} \\ c_B^{(k+1)} \end{bmatrix} = \begin{bmatrix} c_A^{(k)} \\ c_B^{(k)} \end{bmatrix} - \frac{1}{Det(J^{(k)})} \cdot \begin{bmatrix} -\frac{F_{in,1}+F_{in,2}}{V_R} - 4k_{1-} \cdot c_B^{(k)} & -4k_{1-} \cdot c_B^{(k)} \\ -4k_{1+} \cdot c_A^{(k)} & -\frac{F_{in,1}+F_{in,2}}{V_R} - 4k_{1+} \cdot c_A^{(k)} \end{bmatrix} \begin{bmatrix} \frac{dc_A}{dt} \\ \frac{dc_B}{dt} \end{bmatrix}^{(k)}$$

Um nun die stationären Werte berechnen zu können werden Startwerte  $c_A^{(0)}$  und  $c_B^{(0)}$  benötigt. Wählt man zum Beispiel die Startwerte  $c_A^{(0)} = 2 \frac{\text{mol}}{\text{l}}$  und  $c_B^{(0)} = 2 \frac{\text{mol}}{\text{l}}$  so erhält man

$$\begin{aligned} \begin{bmatrix} c_A^{(1)} \\ c_B^{(1)} \end{bmatrix} &= \begin{bmatrix} 2 \frac{\text{mol}}{\text{l}} \\ 2 \frac{\text{mol}}{\text{l}} \end{bmatrix} - \frac{1}{19,36 \frac{1}{\text{min}^2}} \cdot \begin{bmatrix} -16,4 \frac{1}{\text{min}} & -16 \frac{1}{\text{min}} \\ -32 \frac{1}{\text{min}} & -32,4 \frac{1}{\text{min}} \end{bmatrix} \begin{bmatrix} -14,9 \frac{\text{mol}}{\text{min} \cdot \text{l}} \\ 15,25 \frac{\text{mol}}{\text{min} \cdot \text{l}} \end{bmatrix} \\ &= \begin{bmatrix} 2 \frac{\text{mol}}{\text{l}} \\ 2 \frac{\text{mol}}{\text{l}} \end{bmatrix} - \begin{bmatrix} 0,0186 \frac{\text{mol}}{\text{l}} \\ -0,8936 \frac{\text{mol}}{\text{l}} \end{bmatrix} \\ &= \begin{bmatrix} 1,9814 \frac{\text{mol}}{\text{l}} \\ 2,8936 \frac{\text{mol}}{\text{l}} \end{bmatrix} \end{aligned}$$

3.) Die Iterationsvorschriften für die einzelnen Verfahren lauten wie folgt:

a) Explizites Eulerverfahren:

$$\begin{bmatrix} c_A^{(k+1)} \\ c_B^{(k+1)} \end{bmatrix} = \begin{bmatrix} c_A^{(k)} \\ c_B^{(k)} \end{bmatrix} + \Delta t \begin{bmatrix} \frac{1}{V_R} \left( F_{\text{in},1} (c_A^{\text{in},1} - c_A^{(k)}) + F_{\text{in},2} (c_A^{\text{in},2} - c_A^{(k)}) \right) - 2k_{1+} \cdot (c_A^{(k)})^2 + 2k_{1-} \cdot (c_B^{(k)})^2 \\ \frac{1}{V_R} \left( -F_{\text{in},1} c_B^{(k)} + F_{\text{in},2} (c_B^{\text{in},2} - c_B^{(k)}) \right) + 2k_{1+} \cdot (c_A^{(k)})^2 - 2k_{1-} \cdot (c_B^{(k)})^2 \end{bmatrix}$$

Implizites Eulerverfahren:

$$\begin{bmatrix} c_A^{(k+1)} \\ c_B^{(k+1)} \end{bmatrix} = \begin{bmatrix} c_A^{(k)} \\ c_B^{(k)} \end{bmatrix} + \Delta t \begin{bmatrix} \frac{1}{V_R} \left( F_{\text{in},1} (c_A^{\text{in},1} - c_A^{(k+1)}) + F_{\text{in},2} (c_A^{\text{in},2} - c_A^{(k+1)}) \right) - 2k_{1+} \cdot (c_A^{(k+1)})^2 + 2k_{1-} \cdot (c_B^{(k+1)})^2 \\ \frac{1}{V_R} \left( -F_{\text{in},1} c_B^{(k+1)} + F_{\text{in},2} (c_B^{\text{in},2} - c_B^{(k+1)}) \right) + 2k_{1+} \cdot (c_A^{(k+1)})^2 - 2k_{1-} \cdot (c_B^{(k+1)})^2 \end{bmatrix}$$

b) Runge-Kutta-Verfahren 2. Ordnung:

$$(\vec{c})^{(k+1)} = (\vec{c})^{(k)} + \Delta t \cdot m_2$$

mit

$$m_2 = \vec{f} \left( (\vec{c})^{(k)} + \frac{\Delta t}{2} \cdot m_1 \right)$$

und

$$\begin{aligned} m_1 &= \vec{f} \left( (\vec{c})^{(k)} \right) \\ &= \begin{bmatrix} \frac{1}{V_R} \left( F_{\text{in},1} (c_A^{\text{in},1} - c_A^{(k)}) + F_{\text{in},2} (c_A^{\text{in},2} - c_A^{(k)}) \right) - 2k_{1+} \cdot (c_A^{(k)})^2 + 2k_{1-} \cdot (c_B^{(k)})^2 \\ \frac{1}{V_R} \left( -F_{\text{in},1} c_B^{(k)} + F_{\text{in},2} (c_B^{\text{in},2} - c_B^{(k)}) \right) + 2k_{1+} \cdot (c_A^{(k)})^2 - 2k_{1-} \cdot (c_B^{(k)})^2 \end{bmatrix} \end{aligned}$$

Jede der drei Methoden zur numerischen Integration birgt Vor- und Nachteile in sich. Zwar ist der numerische Aufwand beim expliziten Eulerverfahren am geringsten, es ist aber nur für kleine Iterationsschritte  $\Delta t$  stabil.

Das implizite Eulerverfahren hingegen ist auch für große  $\Delta t$  stabil, erfordert aber erheblich höheren Rechenaufwand, da bei jedem Schritt zusätzlich der Wert  $c^{(k+1)}$  iterativ (z.B. mittels Newton-Verfahren) bestimmt werden muss.

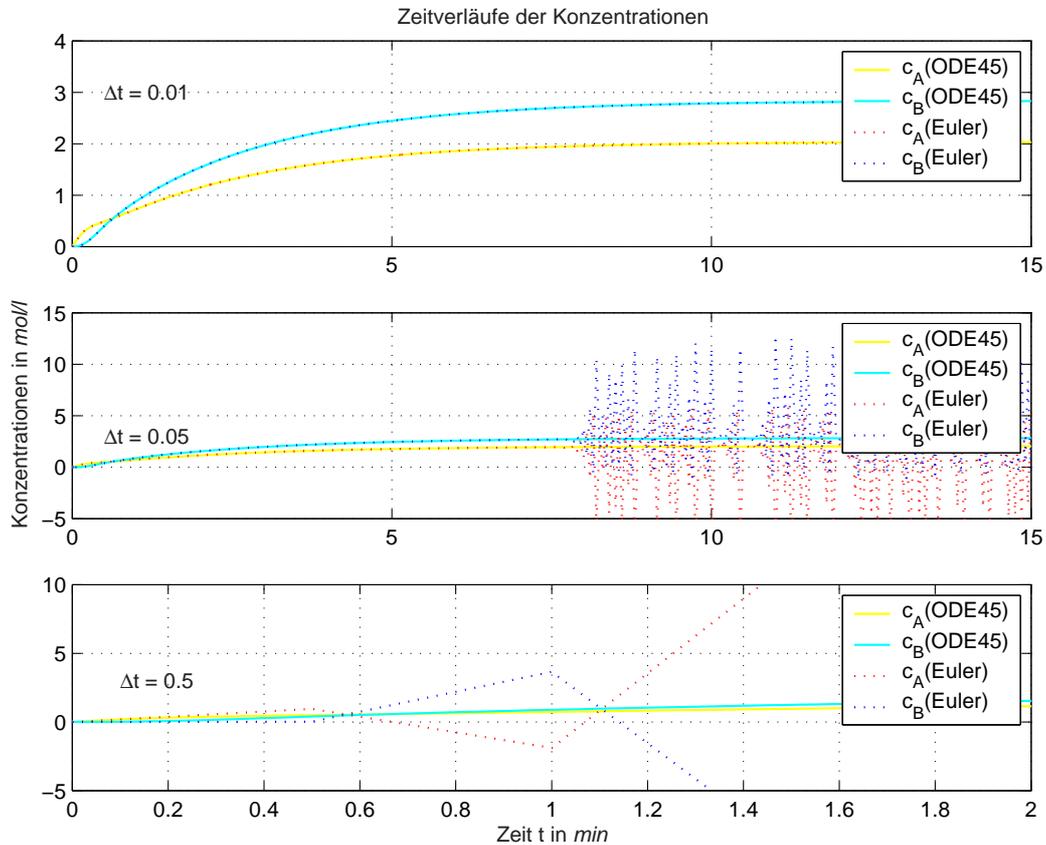
Das Runge-Kutta-Verfahren ist ebenfalls eine explizite Iterationsvorschrift. Ihr Vorteil gegenüber dem expliziten Eulerverfahren liegt aber darin, dass ein gewichtetes Mittel der Anstiege zwischen 2 Punkten zur Berechnung verwendet wird. Die Runge-Kutta-Verfahren höherer Ordnung (z.B. 4. Ordnung) sind zwar rechenaufwendiger aber dafür wiederum stabiler als die niedriger Ordnung (z.B. 2. Ordnung).

4.) Die zeitlichen Verläufe der Konzentrationen im Reaktor wurden numerisch mittels dem expliziten Eulerverfahren mit verschiedenen Schrittweiten iterativ berechnet. Es zeigte sich, dass das Verfahren nur bei der Schrittweite  $\Delta t = 0,01\text{min}$  stabil ist und die errechneten Konzentrationsverläufe mit den Werten der 'ODE45' von MATLAB übereinstimmen.

Bei einer Schrittweite von  $\Delta t = 0,05\text{min}$  liefert das explizite Eulerverfahren bis zum Zeitpunkt  $t \simeq 7\text{min}$  zuverlässige Werte, die mit den Berechnungen der 'ODE45' in guter Näherung übereinstimmen. Danach ändert sich dies und das Verfahren liefert um den stationären Zustand mit hoher Frequenz oszillierende Konzentrationsverläufe. Außerdem werden negative und extrem hohe Werte errechnet, die aus rein logischer Betrachtung nicht erreicht werden können.

Wird eine Schrittweite von  $\Delta t = 0,5\text{min}$  verwendet, zeigt sich bereits nach wenigen Iterationsschritten ein sehr instabiles Verhalten und die errechneten Werte und Konzentrationsverläufe entsprechen in keiner Weise denen der 'ODE45'-Funktion.

Bei der Wahl der Schrittweite muss darauf geachtet werden, dass diese so klein wie nötig aber auch so groß wie möglich gewählt wird. Zum Einen wird dadurch die benötigte Stabilität erreicht und zum Anderen der Rechenaufwand minimiert.



```

%+-----+
%|   Steffi Klinge           STK 03           XXXXXX   |
%+-----+

%function ha5_1      HAUPTPROGRAMM

clear all;
close all;
clc;

titles = 1;

%-----
% VARIABLEN

p.k_1 = 4;           %l/(mol*min)
p.k_2 = 2;           %l/(mol*min)
p.F_in1 = 1.5;       %l/min
p.F_in2 = 0.5;       %l/min
p.c_A_in1 = 5;       %mol/l
p.c_A_in2 = 4;       %mol/l
p.c_B_in2 = 0.5;     %mol/l

```

```

p.V = 5; %1
p.c_vgl = [1000 1000]';

%-----
% ANFANGSWERTE
p.c_0 = [0 0]'; %anfängskonzentrationen
c_0 = [0 0]';
t_end = 15;

%-----
%NEWTON

while (p.c_vgl(1,1)-p.c_0(1,1))^2 > 1e-8 | (p.c_vgl(2,1)-p.c_0(2,1))^2 > 1e-8
    f = [1/p.V*(p.F_in1*(p.c_A_in1-p.c_0(1,1))+p.F_in2*(p.c_A_in2-p.c_0(1,1)))
        -2*p.k_1*(p.c_0(1,1))^2+2*p.k_2*(p.c_0(2,1))^2,
        1/p.V*(p.F_in1*(-p.c_0(2,1))+p.F_in2*(p.c_B_in2-p.c_0(2,1)))
        +2*p.k_1*(p.c_0(1,1))^2-2*p.k_2*(p.c_0(2,1))^2];
    J_inv = 1/((p.F_in1+p.F_in2)^2/p.V^2+4*(p.F_in1+p.F_in2)/p.V*(p.k_1*p.c_0(1,1)+p.k_2*p.c_0(2,1)))*
        [-(p.F_in1+p.F_in2)/p.V-4*p.k_2*p.c_0(2,1) -4*p.k_2*p.c_0(2,1),
        -4*p.k_1*p.c_0(1,1) -(p.F_in1+p.F_in2)/p.V-4*p.k_1*p.c_0(1,1)];
    p.c_neu = p.c_0 - J_inv * f;
    p.c_vgl = p.c_0;
    p.c_0 = p.c_neu;
end

p.c_ss = p.c_neu; %ss-wert speichern

disp(sprintf('===== \n \n NEWTON \n'));
disp(sprintf('Ergebnis: c_A,ss = %0.5g', p.c_neu(1)));
disp(sprintf(' c_B,ss = %0.5g \n', p.c_neu(2)));

%-----
%EXPLIZITER EULER

for i = 1:3
    j=0;
    switch i
        case 1
            delta_t = 0.01; %min
            disp(sprintf('===== \n \n
                ndelta_t = %0.5g \n', delta_t));
        case 2
            delta_t = 0.05; %min
            disp(sprintf('===== \n \n
                ndelta_t = %0.5g \n', delta_t));
        otherwise
            delta_t = 0.5; %min
            disp(sprintf('===== \n \n
                ndelta_t = %0.5g \n', delta_t));
    end
    p.c_0 = [0 0]';
    while (p.c_ss(1,1)-p.c_0(1,1))^2 > 1e-4 | (p.c_ss(2,1)-p.c_0(2,1))^2 > 1e-4
        j = j + 1;
        c_ges_A(i,j) = p.c_0(1,1);
        c_ges_B(i,j) = p.c_0(2,1);
        time(i,j) = delta_t * (j-1);
        f = [1/p.V*(p.F_in1*(p.c_A_in1-p.c_0(1,1))+p.F_in2*(p.c_A_in2-p.c_0(1,1)))
            -2*p.k_1*(p.c_0(1,1))^2+2*p.k_2*(p.c_0(2,1))^2,
            1/p.V*(p.F_in1*(-p.c_0(2,1))+p.F_in2*(p.c_B_in2-p.c_0(2,1)))
        ];
    end
end

```

```

        +2*p.k_1*(p.c_0(1,1))^2-2*p.k_2*(p.c_0(2,1))^2];
p.c_neu = p.c_0 + delta_t * f;
p.c_0 = p.c_neu;
if p.c_neu(1,1) < -15 | p.c_neu(1,1) >30 | p.c_neu(2,1) < -15 |
    p.c_neu(2,1) >30 | time (i,j)>14.99
    break %abbruch bei instabilitaet und oszillation
end
end
disp(sprintf('Iterationen:      %i\n',j));
disp(sprintf('Ergebnis:  c_A = %0.5g',p.c_neu(1)));
disp(sprintf('              c_B = %0.5g\n',p.c_neu(2)));
end

%-----
%ODE 45
[t,c] = ode45(@ha5_2,[0 t_end],c_0, '',p);

%-----
%GRAFISCHE AUSGABE

for i=1:3
    for j=2:max(size(c_ges_A))      %alle nullen durch nan ersetzen für plot
        if c_ges_A(i,j) == 0
            c_ges_A(i,j) = nan;
        end
        if c_ges_B(i,j) == 0
            c_ges_B(i,j) = nan;
        end
    end
end
end

figure (1)
subplot (3,1,1)
plot(t,c(:,1),'y',t,c(:,2),'c', time(1,:), c_ges_A(1,:),'r:',time(1,:), c_ges_B(1,:),
    'b:', 'linewidth',1);
legend('c_A(ODE45)', 'c_B(ODE45)', 'c_A(Euler)', 'c_B(Euler)');
if titles, title('Zeitverläufe der Konzentrationen'); end
grid on;
temp = axis; axis ([temp(1) temp(2) 0 4]);
text(0.5,3,'Deltat = 0.01');
% speichert automatisch gewählte achsen-grenzen, übernimmt davon
% xmin und xmax.      ymin und ymax werden manuell gesetzt.

subplot (3,1,2)
plot(t,c(:,1),'y',t,c(:,2),'c', time(2,:), c_ges_A(2,:),'r:',time(2,:), c_ges_B(2,:),
    'b:', 'linewidth',1);
legend('c_A(ODE45)', 'c_B(ODE45)', 'c_A(Euler)', 'c_B(Euler)');
text(0.5,3,'Deltat = 0.05'); ylabel('Konzentrationen in \it{mol/l}');
grid on; temp = axis; axis ([temp(1) temp(2) -5 15]);

subplot (3,1,3)
plot(t,c(:,1),'y',t,c(:,2),'c', time(3,:), c_ges_A(3,:),'r:',time(3,:), c_ges_B(3,:),
    'b:', 'linewidth',1);
legend('c_A(ODE45)', 'c_B(ODE45)', 'c_A(Euler)', 'c_B(Euler)');
text(0.1,3,'Deltat = 0.5');
grid on; temp = axis; axis ([temp(1) 2 -5 10]);
xlabel('Zeit t in \it{min}');

```

```
%function ha5_2      HILFSPROGRAMM
```

```
function dc = ha5_2(t,c,p);
```

```
r = p.k_1 * c(1)^2 - p.k_2 * (c(2))^2;
```

```
dc(1) = 1/p.V * (p.F_in1 * (p.c_A_in1 - c(1) ) + p.F_in2 * (p.c_A_in2 - c(1) )) - 2*r;
```

```
dc(2) = 1/p.V * (p.F_in1 * ( - c(2)) + p.F_in2 * (p.c_B_in2 - c(2))) + 2*r;
```

```
dc = dc';
```

5.) Da das vorliegende Prozessmodell nichtlinear ist, muss es um den stationären Punkt linearisiert werden um es in die Zustandsraumdarstellung zu überführen.

Im vorliegenden Modell werden  $F_{in,1}$  und  $c_A^{in,1}$  als veränderlich und somit als Eingangsgrößen  $u$  des Systems angenommen.  $F_{in,2}$ ,  $c_A^{in,2}$  und  $c_B^{in,2}$  sind konstant und werden als Parameter des Systems betrachtet. Als Ausgangsgröße  $y$  soll  $c_B$  überwacht werden.  $c_A$  und  $c_B$  sind Zustände  $x$  des Systems. Die allgemeine Form der Zustandsraumdarstellung lautet

$$\Delta \dot{x} = A\Delta x + B\Delta u \quad \Delta y = C\Delta x + D\Delta u$$

mit

$$A = \left[ \frac{\partial f_i}{\partial x_j} \Big|_{x_{ss}, u_{ss}} \right], \quad B = \left[ \frac{\partial f_i}{\partial u_j} \Big|_{x_{ss}, u_{ss}} \right], \quad C = \left[ \frac{\partial g_i}{\partial x_j} \Big|_{x_{ss}, u_{ss}} \right] \quad \text{und} \quad D = \left[ \frac{\partial g_i}{\partial u_j} \Big|_{x_{ss}, u_{ss}} \right]$$

Im folgenden soll vereinfachend die Schreibweise  $\Delta x = x$  für alle Abstandsvariablen verwendet werden.

$$\begin{bmatrix} \dot{c}_A \\ \dot{c}_B \end{bmatrix} = \begin{bmatrix} -\frac{F_{in,1,ss} + F_{in,2}}{V_R} - 4k_{1+} \cdot c_{A,ss} & 4k_{1-} \cdot c_{B,ss} \\ 4k_{1+} \cdot c_{A,ss} & -\frac{F_{in,1,ss} + F_{in,2}}{V_R} - 4k_{1-} \cdot c_{B,ss} \end{bmatrix} \cdot \begin{bmatrix} c_A \\ c_B \end{bmatrix} + \begin{bmatrix} \frac{c_{A,ss}^{in,1} - c_{A,ss}}{V_R} & \frac{F_{in,1,ss}}{V_R} \\ -\frac{c_{B,ss}}{V_R} & 0 \end{bmatrix} \cdot \begin{bmatrix} F_{in,1} \\ c_A^{in,1} \end{bmatrix}$$

$$[c_B] = [0 \quad 1] \cdot \begin{bmatrix} c_A \\ c_B \end{bmatrix} + [0 \quad 0] \cdot \begin{bmatrix} F_{in,1} \\ c_A^{in,1} \end{bmatrix}$$