Define Fed-Batch ODE System

In[∘]:=
```
FB = {
    D[XV[τ], τ] == μ * XV[τ], XV[0] == X0 V0,
    D[PV[τ], τ] == πp * XV[τ], PV[0] == 0,
    D[GV[τ], τ] == F[τ] * (Sf - GV[τ]) - σ * XV[τ], GV[0] == 0,
    D[V[τ], τ] == F[τ], V[0] == V0
   };
```

Normalization rules

In[∘]:=
```
norm = {
    V[τ] → V0 v[τ],
    V[0] → V0 v[0],
    V'[τ] → V0 v'[τ],

    XV[τ] → V0 Sf Yxs xv[τ],
    XV[0] → V0 Sf Yxs xv[0],
    XV'[τ] → V0 Sf Yxs xv'[τ],
    X0 → x0 Sf Yxs,

    PV[τ] → V0 Sf Yxs pv[τ],
    PV[0] → V0 Sf Yxs pv[0],
    PV'[τ] → V0 Sf Yxs pv'[τ],

    GV[τ] → V0 Sf Yxs gv[τ],
    GV[0] → V0 Sf Yxs gv[0],
    GV'[τ] → V0 Sf Yxs gv'[τ],

    F[τ] → F0 f[τ],
    Finf → F0 Finf,

    μ → F0 / V0 μ,
    μf → F0 / V0 μf,
    π0 → F0 / V0 π0,
    ρ → F0 / V0 ρ
   };
```

Apply normalization and set glucose mass balance to zero

In[∘]:=
```
FBnorm = Simplify[FB /. norm, {Sf > 0, V0 > 0, Yxs > 0}];
FBnormg0 = Simplify[FBnorm /. gv'[τ] → 0 /. gv[τ] → 0];
```

Solve for the feed rate

In[∘]:=
```
sol = Solve[Part[FBnormg0, 5], f[τ]]
```

Out[∘]=

$$\left\{\left\{f[\tau] \to \frac{V0\ Yxs\ \sigma\ xv[\tau]}{F0}\right\}\right\}$$

Setup substrate consumption rate and production rate

In[∘]:=
```
sigmaglc = f[τ] == (f[τ] /. sol[[1]])
sigmamu = σ → (μ / Yxs + πp / Yps + ρ / Yas) /. norm
πp = π1 * μ + π0 /. norm
```

Out[∘]=

$$f[\tau] == \frac{V0\ Yxs\ \sigma\ xv[\tau]}{F0}$$

Out[∘]=

$$\sigma \to \frac{F0\ \mu}{V0\ Yxs} + \frac{\pi p}{Yps} + \frac{F0\ \rho}{V0\ Yas}$$

Out[∘]=

$$\frac{F0\ \pi 0}{V0} + \frac{F0\ \mu\ \pi 1}{V0}$$

Solve for the growth rate μ

In[∘]:=
```
mu = Apart[Solve[sigmaglc /. sigmamu, μ]]
mualpha = ExpandAll[mu /. (1 / (Yps + π1 Yxs)) → α / Yps]
{mu} = Simplify[mualpha /. π0 Yxs / Yps → β - ρ Yxs / Yas]
```

Out[∘]=

$$\left\{\left\{\mu \to -\frac{Yxs\ (Yas\ \pi 0 + Yps\ \rho)}{Yas\ (Yps + Yxs\ \pi 1)} + \frac{Yps\ f[\tau]}{(Yps + Yxs\ \pi 1)\ xv[\tau]}\right\}\right\}$$

Out[∘]=

$$\left\{\left\{\mu \to -\frac{Yxs\ \alpha\ \pi 0}{Yps} - \frac{Yxs\ \alpha\ \rho}{Yas} + \frac{\alpha\ f[\tau]}{xv[\tau]}\right\}\right\}$$

Out[∘]=

$$\left\{\left\{\mu \to -\alpha\ \beta + \frac{\alpha\ f[\tau]}{xv[\tau]}\right\}\right\}$$

Feed Rates

Define feed rates and apply normalization

In[ ]:=
```
logfeed = F[τ] → Finf / (1 + Exp[-μf τ] (Finf / F0 - 1)) /. norm
expfeed = F[τ] → Limit[logfeed[[2]], Finf → Infinity];
linfeed = F[τ] → Limit[Limit[logfeed[[2]], Finf → Infinity], μf → 0];
feed = logfeed /. F0 f_ → f
```

Out[ ]=

$$F0\, f[\tau] \to \frac{F0\, Finf}{1 + e^{-\frac{F0\, \mu f\, \tau}{V0}} (-1 + Finf)}$$

Out[ ]=

$$f[\tau] \to \frac{Finf}{1 + e^{-\frac{F0\, \mu f\, \tau}{V0}} (-1 + Finf)}$$

Solve ODE System

Solve ODEs, use DSolveChangeVariables to solve by substitution

In[ ]:=
```
odexpv = Flatten[FullSimplify[Join[{FBnormg0[[1 ;; 2]] /. mu} /. feed,
    {FBnormg0[[3 ;; 4]] /. mu} /. feed, FBnormg0[[7 ;; 8]] /. feed]]];
DSolveChangeVariables[Inactive[DSolve][odexpv, {xv, pv, v}, τ],
  {xvnew, pvnew, vnew}, t, t == τ F0 / V0];
res = Activate[%];
xvnew = xvnew /. First[res][[2]];
pvnew = pvnew /. First[res][[1]];
vnew = vnew /. First[res][[3]];
xv[t_] = xvnew[t]
pv[t_] = pvnew[t]
v[t_] = vnew[t];
x[t_] = xv[t] / v[t];
```

Out[ ]=

$$\frac{1}{(-1+\text{Finf})\ (\alpha\beta+\mu f)} e^{-t\alpha\beta}\Big(-x0\,\alpha\,\beta+\text{Finf}\,x0\,\alpha\,\beta-x0\,\mu f+\text{Finf}\,x0\,\mu f-\text{Finf}\,\alpha\,\text{Hypergeometric2F1}\Big[1,1+\frac{\alpha\beta}{\mu f},2+\frac{\alpha\beta}{\mu f},-\frac{1}{-1+\text{Finf}}\Big]+e^{t\alpha\beta+t\mu f}\,\text{Finf}\,\alpha\,\text{Hypergeometric2F1}\Big[1,1+\frac{\alpha\beta}{\mu f},2+\frac{\alpha\beta}{\mu f},-\frac{e^{t\mu f}}{-1+\text{Finf}}\Big]\Big)$$

Out[ ]=

$$\frac{1}{(-1+\text{Finf})\ \alpha\,\beta\,\mu f\ (\alpha\beta+\mu f)} e^{-t\alpha\beta}\Big(x0\,\alpha\,\beta\,\mu f\,\pi 0-e^{t\alpha\beta}\,x0\,\alpha\,\beta\,\mu f\,\pi 0-\text{Finf}\,x0\,\alpha\,\beta\,\mu f\,\pi 0+e^{t\alpha\beta}\,\text{Finf}\,x0\,\alpha\,\beta\,\mu f\,\pi 0+x0\,\mu f^2\,\pi 0-e^{t\alpha\beta}\,x0\,\mu f^2\,\pi 0-\text{Finf}\,x0\,\mu f^2\,\pi 0+e^{t\alpha\beta}\,\text{Finf}\,x0\,\mu f^2\,\pi 0-x0\,\alpha^2\,\beta^2\,\mu f\,\pi 1+e^{t\alpha\beta}\,x0\,\alpha^2\,\beta^2\,\mu f\,\pi 1+\text{Finf}\,x0\,\alpha^2\,\beta^2\,\mu f\,\pi 1-e^{t\alpha\beta}\,\text{Finf}\,x0\,\alpha^2\,\beta^2\,\mu f\,\pi 1-x0\,\alpha\,\beta\,\mu f^2\,\pi 1+e^{t\alpha\beta}\,x0\,\alpha\,\beta\,\mu f^2\,\pi 1+\text{Finf}\,x0\,\alpha\,\beta\,\mu f^2\,\pi 1-e^{t\alpha\beta}\,\text{Finf}\,x0\,\alpha\,\beta\,\mu f^2\,\pi 1+\text{Finf}\,\alpha\,\mu f\,\pi 0\,\text{Hypergeometric2F1}\Big[1,1+\frac{\alpha\beta}{\mu f},2+\frac{\alpha\beta}{\mu f},-\frac{1}{-1+\text{Finf}}\Big]-\text{Finf}\,\alpha^2\,\beta\,\mu f\,\pi 1\,\text{Hypergeometric2F1}\Big[1,1+\frac{\alpha\beta}{\mu f},2+\frac{\alpha\beta}{\mu f},-\frac{1}{-1+\text{Finf}}\Big]-e^{t\alpha\beta+t\mu f}\,\text{Finf}\,\alpha\,\mu f\,\pi 0\,\text{Hypergeometric2F1}\Big[1,1+\frac{\alpha\beta}{\mu f},2+\frac{\alpha\beta}{\mu f},-\frac{e^{t\mu f}}{-1+\text{Finf}}\Big]+e^{t\alpha\beta+t\mu f}\,\text{Finf}\,\alpha^2\,\beta\,\mu f\,\pi 1\,\text{Hypergeometric2F1}\Big[1,1+\frac{\alpha\beta}{\mu f},2+\frac{\alpha\beta}{\mu f},-\frac{e^{t\mu f}}{-1+\text{Finf}}\Big]+e^{t\alpha\beta}\,\text{Finf}\,\alpha^2\,\beta\,\pi 0\,\text{Log}[\text{Finf}]-e^{t\alpha\beta}\,\text{Finf}^2\,\alpha^2\,\beta\,\pi 0\,\text{Log}[\text{Finf}]+e^{t\alpha\beta}\,\text{Finf}\,\alpha\,\mu f\,\pi 0\,\text{Log}[\text{Finf}]-e^{t\alpha\beta}\,\text{Finf}^2\,\alpha\,\mu f\,\pi 0\,\text{Log}[\text{Finf}]-e^{t\alpha\beta}\,\text{Finf}\,\alpha^2\,\beta\,\pi 0\,\text{Log}\big[-1+e^{t\mu f}+\text{Finf}\big]+e^{t\alpha\beta}\,\text{Finf}^2\,\alpha^2\,\beta\,\pi 0\,\text{Log}\big[-1+e^{t\mu f}+\text{Finf}\big]-e^{t\alpha\beta}\,\text{Finf}\,\alpha\,\mu f\,\pi 0\,\text{Log}\big[-1+e^{t\mu f}+\text{Finf}\big]+e^{t\alpha\beta}\,\text{Finf}^2\,\alpha\,\mu f\,\pi 0\,\text{Log}\big[-1+e^{t\mu f}+\text{Finf}\big]\Big)$$

Simplify product and volume solutions

In[∘]:=
```
logTerms = Select[Collect[Expand[pv[t]], Log], ! FreeQ[#, Log] &]
FullSimplify[logTerms]
pv01[t_] = Expand[pv[t]] /. logTerms → %
```

Out[∘]=

$$\frac{\text{Finf}\,\pi 0\,\text{Log}[\text{Finf}]}{(-1+\text{Finf})\,\beta\,(\alpha\,\beta+\mu f)} - \frac{\text{Finf}^2\,\pi 0\,\text{Log}[\text{Finf}]}{(-1+\text{Finf})\,\beta\,(\alpha\,\beta+\mu f)} + \frac{\text{Finf}\,\alpha\,\pi 0\,\text{Log}[\text{Finf}]}{(-1+\text{Finf})\,\mu f\,(\alpha\,\beta+\mu f)} - \frac{\text{Finf}^2\,\alpha\,\pi 0\,\text{Log}[\text{Finf}]}{(-1+\text{Finf})\,\mu f\,(\alpha\,\beta+\mu f)} - \frac{\text{Finf}\,\pi 0\,\text{Log}\left[-1+e^{t\,\mu f}+\text{Finf}\right]}{(-1+\text{Finf})\,\beta\,(\alpha\,\beta+\mu f)} + \frac{\text{Finf}^2\,\pi 0\,\text{Log}\left[-1+e^{t\,\mu f}+\text{Finf}\right]}{(-1+\text{Finf})\,\beta\,(\alpha\,\beta+\mu f)} - \frac{\text{Finf}\,\alpha\,\pi 0\,\text{Log}\left[-1+e^{t\,\mu f}+\text{Finf}\right]}{(-1+\text{Finf})\,\mu f\,(\alpha\,\beta+\mu f)} + \frac{\text{Finf}^2\,\alpha\,\pi 0\,\text{Log}\left[-1+e^{t\,\mu f}+\text{Finf}\right]}{(-1+\text{Finf})\,\mu f\,(\alpha\,\beta+\mu f)}$$

Out[∘]=

$$\frac{\text{Finf}\,\pi 0\,\left(-\text{Log}[\text{Finf}]+\text{Log}\left[-1+e^{t\,\mu f}+\text{Finf}\right]\right)}{\beta\,\mu f}$$

Out[∘]=

$$-\frac{x0\,\pi 0}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} + \frac{e^{-t\,\alpha\,\beta}\,x0\,\pi 0}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} + \frac{\text{Finf}\,x0\,\pi 0}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} - \frac{e^{-t\,\alpha\,\beta}\,\text{Finf}\,x0\,\pi 0}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} - \frac{x0\,\mu f\,\pi 0}{(-1+\text{Finf})\,\alpha\,\beta\,(\alpha\,\beta+\mu f)} + \frac{e^{-t\,\alpha\,\beta}\,x0\,\mu f\,\pi 0}{(-1+\text{Finf})\,\alpha\,\beta\,(\alpha\,\beta+\mu f)} + \frac{\text{Finf}\,x0\,\mu f\,\pi 0}{(-1+\text{Finf})\,\alpha\,\beta\,(\alpha\,\beta+\mu f)} - \frac{e^{-t\,\alpha\,\beta}\,\text{Finf}\,x0\,\mu f\,\pi 0}{(-1+\text{Finf})\,\alpha\,\beta\,(\alpha\,\beta+\mu f)} + \frac{x0\,\alpha\,\beta\,\pi 1}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} - \frac{e^{-t\,\alpha\,\beta}\,x0\,\alpha\,\beta\,\pi 1}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} - \frac{\text{Finf}\,x0\,\alpha\,\beta\,\pi 1}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} + \frac{e^{-t\,\alpha\,\beta}\,\text{Finf}\,x0\,\alpha\,\beta\,\pi 1}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} + \frac{x0\,\mu f\,\pi 1}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} - \frac{e^{-t\,\alpha\,\beta}\,x0\,\mu f\,\pi 1}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} - \frac{\text{Finf}\,x0\,\mu f\,\pi 1}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} + \frac{e^{-t\,\alpha\,\beta}\,\text{Finf}\,x0\,\mu f\,\pi 1}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} + \frac{e^{-t\,\alpha\,\beta}\,\text{Finf}\,\pi 0\,\text{Hypergeometric2F1}\left[1,1+\frac{\alpha\,\beta}{\mu f},2+\frac{\alpha\,\beta}{\mu f},-\frac{1}{-1+\text{Finf}}\right]}{(-1+\text{Finf})\,\beta\,(\alpha\,\beta+\mu f)} - \frac{e^{-t\,\alpha\,\beta}\,\text{Finf}\,\alpha\,\pi 1\,\text{Hypergeometric2F1}\left[1,1+\frac{\alpha\,\beta}{\mu f},2+\frac{\alpha\,\beta}{\mu f},-\frac{1}{-1+\text{Finf}}\right]}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} - \frac{e^{t\,\mu f}\,\text{Finf}\,\pi 0\,\text{Hypergeometric2F1}\left[1,1+\frac{\alpha\,\beta}{\mu f},2+\frac{\alpha\,\beta}{\mu f},-\frac{e^{t\,\mu f}}{-1+\text{Finf}}\right]}{(-1+\text{Finf})\,\beta\,(\alpha\,\beta+\mu f)} + \frac{e^{t\,\mu f}\,\text{Finf}\,\alpha\,\pi 1\,\text{Hypergeometric2F1}\left[1,1+\frac{\alpha\,\beta}{\mu f},2+\frac{\alpha\,\beta}{\mu f},-\frac{e^{t\,\mu f}}{-1+\text{Finf}}\right]}{(-1+\text{Finf})\,(\alpha\,\beta+\mu f)} + \frac{\text{Finf}\,\pi 0\,\left(-\text{Log}[\text{Finf}]+\text{Log}\left[-1+e^{t\,\mu f}+\text{Finf}\right]\right)}{\beta\,\mu f}$$

Apply log(a)-log(b)=log(a/b) to product

In[ ]:=
```
pv02[t_] = Simplify[pv01[t] /.
    (-Log[Finf] + Log[-1 + ⅇ^(t μf) + Finf]) → Log[(Exp[μf t] - 1) / Finf + 1]];
pv[t_] = pv02[t];
p[t_] = pv[t] / v[t]
```

Out[ ]=

$$\left(e^{-t\alpha\beta}\left(\text{Finf}\,\alpha\,\mu f\,(\pi 0-\alpha\,\beta\,\pi 1)\,\text{Hypergeometric2F1}\left[1,\,1+\frac{\alpha\,\beta}{\mu f},\,2+\frac{\alpha\,\beta}{\mu f},\,\frac{1}{1-\text{Finf}}\right]+e^{t\,(\alpha\,\beta+\mu f)}\,\text{Finf}\,\alpha\,\mu f\,(-\pi 0+\alpha\,\beta\,\pi 1)\,\text{Hypergeometric2F1}\left[1,\,1+\frac{\alpha\,\beta}{\mu f},\,2+\frac{\alpha\,\beta}{\mu f},\,-\frac{e^{t\,\mu f}}{-1+\text{Finf}}\right]+(-1+\text{Finf})\,(\alpha\,\beta+\mu f)\left(\left(-1+e^{t\,\alpha\,\beta}\right)\,x0\,\mu f\,(\pi 0-\alpha\,\beta\,\pi 1)+e^{t\,\alpha\,\beta}\,\text{Finf}\,\alpha\,\pi 0\,\text{Log}\left[\frac{-1+e^{t\,\mu f}+\text{Finf}}{\text{Finf}}\right]\right)\right)\right)\Big/\left((-1+\text{Finf})\,\alpha\,\beta\,(\alpha\,\beta+\mu f)\,\left(\mu f-\text{Finf}\,\text{Log}[\text{Finf}]+\text{Finf}\,\text{Log}\left[-1+e^{t\,\mu f}+\text{Finf}\right]\right)\right)$$

Apply log (a) - log (b) = log (a/b) to volume

In[ ]:=
```
v[t_] = Simplify[v[t] /. (-Finf Log[Finf] + Finf Log[-1 + ⅇ^(t μf) + Finf]) →
    Finf Log[(Exp[μf t] - 1) / Finf + 1]]
```

Out[ ]=

$$1+\frac{\text{Finf}\,\text{Log}\left[\frac{-1+e^{t\,\mu f}+\text{Finf}}{\text{Finf}}\right]}{\mu f}$$

Exponential Feed

Define functions for exponential feed by setting Finf to infinitiy

In[∘]:=
```
vexp[t_] = Limit[v[t], Finf → Infinity]
xvexp[t_] = Limit[xv[t], Finf → Infinity]
xexp[t_] = xvexp[t] / vexp[t]
pvexp[t_] = Limit[pv[t], Finf → Infinity];
pexp[t_] = pvexp[t] / vexp[t]
```

Out[∘]=

$$\frac{-1+e^{t\,\mu f}+\mu f}{\mu f}$$

Out[∘]=

$$\frac{e^{-t\,\alpha\,\beta}\left(\alpha\left(-1+e^{t\,(\alpha\,\beta+\mu f)}+x0\,\beta\right)+x0\,\mu f\right)}{\alpha\,\beta+\mu f}$$

Out[∘]=

$$\frac{e^{-t\,\alpha\,\beta}\,\mu f\left(\alpha\left(-1+e^{t\,(\alpha\,\beta+\mu f)}+x0\,\beta\right)+x0\,\mu f\right)}{\left(-1+e^{t\,\mu f}+\mu f\right)\,(\alpha\,\beta+\mu f)}$$

Out[∘]=

$$\left(e^{-t\,\alpha\,\beta}\left(-\alpha\,\mu f\,(-\pi 0+\alpha\,\beta\,\pi 1)+e^{t\,(\alpha\,\beta+\mu f)}\,\alpha\,\mu f\,(-\pi 0+\alpha\,\beta\,\pi 1)+(\alpha\,\beta+\mu f)\left(e^{t\,\alpha\,\beta}\left(-1+e^{t\,\mu f}\right)\alpha\,\pi 0+\left(-1+e^{t\,\alpha\,\beta}\right)x0\,\mu f\,(\pi 0-\alpha\,\beta\,\pi 1)\right)\right)\right)\Big/\left(\alpha\,\beta\left(-1+e^{t\,\mu f}+\mu f\right)(\alpha\,\beta+\mu f)\right)$$

Constant Feed

Define functions for constant feed rate by setting Finf to 1.
Use PowerExpand to apply Log(e^x)=x

In[∘]:=
```
vlin[t_] = PowerExpand[v[t] /. Finf → 1]
```

Out[∘]=

$$1+t$$

In[∘]:=
```
FullSimplify[Limit[PowerExpand[xv[t]], Finf → 1, Direction → -1]]
xvlin[t_] = Normal[%];
xlin[t_] = xvlin[t] / vlin[t]
```

Out[∘]=

$$\frac{1+e^{-t\,\alpha\,\beta}\,(-1+x0\,\beta)}{\beta} \quad \text{if} \quad \textit{condition}$$

Out[∘]=

$$\frac{1+e^{-t\,\alpha\,\beta}\,(-1+x0\,\beta)}{(1+t)\,\beta}$$

In[ ]:=
```
PowerExpand[
 FullSimplify[Limit[PowerExpand[pv[t]], Finf → 1, Direction → -1]]]
pvlin[t_] = Normal[%];
plin[t_] = pvlin[t] / vlin[t]
```

Out[ ]=

$$\frac{t\,\beta\,\pi0 + \frac{e^{-t\,\alpha\,\beta}\left(-1+e^{t\,\alpha\,\beta}\right)(-1+x0\,\beta)\,(\pi0-\alpha\,\beta\,\pi1)}{\alpha}}{\beta^2} \quad \text{if condition}$$

Out[ ]=

$$\frac{t\,\beta\,\pi0 + \frac{e^{-t\,\alpha\,\beta}\left(-1+e^{t\,\alpha\,\beta}\right)(-1+x0\,\beta)\,(\pi0-\alpha\,\beta\,\pi1)}{\alpha}}{(1+t)\,\beta^2}$$

Prepare for data creation and export

Setup filenames, data directory and yield values

In[ ]:=
```
datadir = "~/fedbatch/data/";
fileNames = {"x01", "x02", "x03", "x04", "x05", "x06", "x07"};
Yieldxs = Yxs → 0.022;
Yieldpx = Ypx → 69.3;
Yieldas = Yas → 2;
Yieldpa = Ypa → 1;
```

Calculate Product-Substrate Yield

In[ ]:=
```
Yieldps = Yps → Yxs Ypx Yas Ypa /. Yieldxs /. Yieldpx /. Yieldas /. Yieldpa
```

Out[ ]=
```
Yps → 3.0492
```

Use α and β for simplification, define production rates for each case

In[ ]:=
```
α → (1 / (1 + π1 Yxs / Yps)) /. Yieldxs /. Yieldps /. Yieldas
β → (π0 Yxs / Yps + ρ Yxs / Yas) /. Yieldxs /. Yieldps /. Yieldas
```

Out[ ]=

$$\alpha \to \frac{1}{1 + 0.00721501\,\pi1}$$

Out[ ]=

$$\beta \to 0.00721501\,\pi0 + 0.011\,\rho$$

Set alpha1 and beta 1 for α=1 and β=0 (no production)

In[ ]:=
```
α → (1 / (1 + π1 Yxs / Yps)) /. Yieldxs /. Yieldps /. Yieldas /. π1 → 0;
alpha1 = %
β → (π0 Yxs / Yps + ρ Yxs / Yas) /. Yieldxs /. Yieldps /. Yieldas /. π0 → 0 /. ρ → 0;
beta1 = %
```

Out[ ]=
```
α → 1.
```

Out[ ]=
```
β → 0.
```

α=0.8 and β=0

In[ ]:=
```
α → (1 / (1 + π1 Yxs / Yps)) /. Yieldxs /. Yieldps /. Yieldas /. π1 → 30
β → (π0 Yxs / Yps + ρ Yxs / Yas) /. Yieldxs /. Yieldps /. Yieldas /. π0 → 0 /. ρ → 0
```

Out[ ]=
```
α → 0.822064
```

Out[ ]=
```
β → 0.
```

α=1 and β=0.3

In[ ]:=
```
α → (1 / (1 + π1 Yxs / Yps)) /. Yieldxs /. Yieldps /. Yieldas /. π1 → 0
β → (π0 Yxs / Yps + ρ Yxs / Yas) /. Yieldxs /. Yieldps /. Yieldas /. π0 → 20 /. ρ → 10
```

Out[ ]=
```
α → 1.
```

Out[ ]=
```
β → 0.2543
```

Set alpha2 and beta2 for α=0.8 and β=0.3 (growth-coupled and growth-decoupled production)

In[ ]:=
```
α → (1 / (1 + π1 Yxs / Yps)) /. Yieldxs /. Yieldps /. Yieldas /. π1 → 30;
alpha2 = %
β → (π0 Yxs / Yps + ρ Yxs / Yas) /. Yieldxs /. Yieldps /. Yieldas /. π0 → 20 /.
  ρ → 10;
beta2 = %
```

Out[ ]=
```
α → 0.822064
```

Out[ ]=
```
β → 0.2543
```

Set production rates for each case

In[ ]:=
```
pi01 = π0 → 0;
pi02 = π0 → 20;
pi11 = π1 → 0;
pi12 = π1 → 30;
rho = ρ → 10;
```

In[ ]:=
```
(* Vary values of initial biomass concentration *)
xValues = {x1, x2, x3, x4, x5, x6, x7};
x1 = x0 → 0.001;
x2 = x0 → 0.0022;
x3 = x0 → 0.0047;
x4 = x0 → 0.01;
x5 = x0 → 0.022;
x6 = x0 → 0.047;
x7 = x0 → 0.1;
```

Create and export data

Volume data for each feed rate from 0-20 t with μf=0.7 and Finf=2

In[ ]:=
```
Vlog = Table[{t, v[t] /. μf → 0.7 /. Finf → 2}, {t, 0, 20, 0.1}];
Vexp = Table[{t, vexp[t] /. μf → 0.7}, {t, 0, 20, 0.1}];
Vlin = Table[{t, vlin[t]}, {t, 0, 20, 0.1}];
VTable =
  Transpose[{Vlog[[All, 1]], Vlog[[All, 2]], Vexp[[All, 2]], Vlin[[All, 2]]}];
Export[FileNameJoin[{ToFileName[datadir], "Vcombined.csv"}], VTable, "CSV"]
```

Out[ ]=
```
~/fedbatch/data/Vcombined.csv
```

Biomass concentration data for each production type and initial biomass concentration

In[ ]:=
```
(*Iterate through the x0 values and export the data for logistic Feed*)
Do[(*Calculate the data for the current x0 value*)
 xlog1 =
  Table[{t, x[t] /. alpha1 /. beta1 /. xValues[[i]] /. μf → 0.7 /. Finf → 2},
   {t, 0, 20, 0.1}];
 xlog2 =
  Table[{t, x[t] /. alpha2 /. beta1 /.  xValues[[i]] /. μf → 0.7 /. Finf → 2},
   {t, 0, 20, 0.1}];
 xlog3 =
  Table[{t, x[t] /. alpha1 /. beta2 /.  xValues[[i]] /. μf → 0.7 /. Finf → 2},
   {t, 0, 20, 0.1}];
 xlog4 =
  Table[{t, x[t] /. alpha2 /. beta2 /.  xValues[[i]] /. μf → 0.7 /. Finf → 2},
   {t, 0, 20, 0.1}];
 (*Prepare the data table*)
 xlogTable =
  Transpose[{xlog1[[All, 1]], xlog1[[All, 2]],
    xlog2[[All, 2]], xlog3[[All, 2]], xlog4[[All, 2]]}];
 (*Export the data to a CSV file*)
 Export[
  datadir <> "xlog_" <> fileNames[[i]] <> "_combined.csv", xlogTable, "CSV"];
 , {i, 1, Length[xValues]}]
```

In[ ]:=
```
(*Iterate through the x0 values and export the data for exponential feed*)
Do[(*Calculate the data for the current x0 value*)
 xexp1 = Table[
   {t, xexp[t] /. alpha1 /. beta1 /. xValues[[i]] /. μf → 0.7}, {t, 0, 20, 0.1}];
 xexp2 = Table[
   {t, xexp[t] /. alpha2 /. beta1 /.  xValues[[i]] /. μf → 0.7}, {t, 0, 20, 0.1}];
 xexp3 = Table[
   {t, xexp[t] /. alpha1 /. beta2 /.  xValues[[i]] /. μf → 0.7}, {t, 0, 20, 0.1}];
 xexp4 = Table[
   {t, xexp[t] /. alpha2 /. beta2 /.  xValues[[i]] /. μf → 0.7}, {t, 0, 20, 0.1}];
 (*Prepare the data table*)
 xexpTable =
  Transpose[{xexp1[[All, 1]], xexp1[[All, 2]],
    xexp2[[All, 2]], xexp3[[All, 2]], xexp4[[All, 2]]}];
 (*Export the data to a CSV file*)
 Export[
  datadir <> "xexp_" <> fileNames[[i]] <> "_combined.csv", xexpTable, "CSV"];
 , {i, 1, Length[xValues]}]
```

In[ ]:=
```
(*Iterate through the x0 values and export the data for constant feed *)
Do[(*Calculate the data for the current x0 value*)
 xlin1 =
  Table[{t, Limit[xlin[t] /. alpha1 /. xValues⟦i⟧, beta1]}, {t, 0, 20, 0.1}];
 xlin2 =
  Table[{t, Limit[xlin[t] /. alpha2 /.  xValues⟦i⟧, beta1]}, {t, 0, 20, 0.1}];
 xlin3 =
  Table[{t, xlin[t] /. alpha1 /. beta2 /.  xValues⟦i⟧}, {t, 0, 20, 0.1}];
 xlin4 =
  Table[{t, xlin[t] /. alpha2 /. beta2 /.  xValues⟦i⟧}, {t, 0, 20, 0.1}];
 (*Prepare the data table*)
 xlinTable =
  Transpose[{xlin1⟦All, 1⟧, xlin1⟦All, 2⟧,
    xlin2⟦All, 2⟧, xlin3⟦All, 2⟧, xlin4⟦All, 2⟧}];
 (*Export the data to a CSV file*)
 Export[
  datadir <> "xlin_" <> fileNames⟦i⟧ <> "_combined.csv", xlinTable, "CSV"];
 , {i, 1, Length[xValues]}]
```

Product concentration data for each production type and initial biomass concentration

In[ ]:=
```
(*Iterate through the x0 values and export the data for logistic feed*)
Do[(*Calculate the data for the current x0 value*)
 plog1 =
  Table[{t, Limit[p[t] /. alpha1 /. xValues⟦i⟧ /. pi01 /. pi11 /. μf → 0.7 /.
       Finf → 2, beta1]}, {t, 0, 20, 0.1}];
 plog2 =
  Table[{t, Limit[p[t] /. alpha2 /.  xValues⟦i⟧ /. pi01 /. pi12 /. μf → 0.7 /.
       Finf → 2, beta1]}, {t, 0, 20, 0.1}];
 plog3 =
  Table[{t, p[t] /. alpha1 /. beta2 /.  xValues⟦i⟧ /. pi02 /. pi11 /. μf → 0.7 /.
      Finf → 2}, {t, 0, 20, 0.1}];
 plog4 =
  Table[{t, p[t] /. alpha2 /. beta2 /.  xValues⟦i⟧ /. pi02 /. pi12 /. μf → 0.7 /.
      Finf → 2}, {t, 0, 20, 0.1}];
 (*Prepare the data table*)
 plogTable =
  Transpose[{plog1⟦All, 1⟧, plog1⟦All, 2⟧,
    plog2⟦All, 2⟧, plog3⟦All, 2⟧, plog4⟦All, 2⟧}];
 (*Export the data to a CSV file*)
 Export[
  datadir <> "plog_" <> fileNames⟦i⟧ <> "_combined.csv", plogTable, "CSV"];
 , {i, 1, Length[xValues]}]
```

In[ ]:=
```
(*Iterate through the x0 values
 and export the data for exponential feed *)
Do[(*Calculate the data for the current x0 value*)
 pexp1 =
  Table[{t, Limit[pexp[t] /. alpha1 /. xValues[[i]] /. pi01 /. pi11 /. μf → 0.7,
     beta1]}, {t, 0, 20, 0.1}];
 pexp2 =
  Table[{t, Limit[pexp[t] /. alpha2 /.  xValues[[i]] /. pi01 /. pi12 /. μf → 0.7,
     beta1]}, {t, 0, 20, 0.1}];
 pexp3 = Table[
   {t, pexp[t] /. alpha1 /. beta2 /.  xValues[[i]] /. pi02 /. pi11 /. μf → 0.7},
   {t, 0, 20, 0.1}];
 pexp4 = Table[
   {t, pexp[t] /. alpha2 /. beta2 /.  xValues[[i]] /. pi02 /. pi12 /. μf → 0.7},
   {t, 0, 20, 0.1}];
 (*Prepare the data table*)
 pexpTable =
  Transpose[{pexp1[[All, 1]], pexp1[[All, 2]],
    pexp2[[All, 2]], pexp3[[All, 2]], pexp4[[All, 2]]}];
 (*Export the data to a CSV file*)
 Export[
  datadir <> "pexp_" <> fileNames[[i]] <> "_combined.csv", pexpTable, "CSV"];
 , {i, 1, Length[xValues]}]
```

In[ ]:=
```
(*Iterate through the x0 values and export the data for constant feed *)
Do[(*Calculate the data for the current x0 value*)
 plin1 =
  Table[{t, Limit[plin[t] /. alpha1 /. xValues[[i]] /. pi01 /. pi11, beta1]},
   {t, 0, 20, 0.1}];
 plin2 =
  Table[{t, Limit[plin[t] /. alpha2 /.  xValues[[i]] /. pi01 /. pi12, beta1]},
   {t, 0, 20, 0.1}];
 plin3 = Table[{t, plin[t] /. alpha1 /. beta2 /.  xValues[[i]] /. pi02 /. pi11},
   {t, 0, 20, 0.1}];
 plin4 = Table[{t, plin[t] /. alpha2 /. beta2 /.  xValues[[i]] /. pi02 /. pi12},
   {t, 0, 20, 0.1}];
 (*Prepare the data table*)
 plinTable =
  Transpose[{plin1[[All, 1]], plin1[[All, 2]],
    plin2[[All, 2]], plin3[[All, 2]], plin4[[All, 2]]}];
 (*Export the data to a CSV file*)
 Export[
  datadir <> "plin_" <> fileNames[[i]] <> "_combined.csv", plinTable, "CSV"];
 , {i, 1, Length[xValues]}]
```

TRY metrics

Define Yield functions. Due to normalization, the volume corresponds directly to the substrate

In[ ]:=
```
Ylog[t_] = p[t] / v[t];
Yexp[t_] = pexp[t] / vexp[t];
Ylin[t_] = plin[t] / vlin[t];
```

In[ ]:=
```
(*Iterate through the x0 values and
 export the yield data for logistic feed*)
Do[(*Calculate the data for the current x0 value*)
 Ylog1 = Table[
   {t, Limit[Ylog[t] /. alpha1 /. xValues⟦i⟧ /. pi01 /. pi11 /. μf → 0.7 /.
      Finf → 2, beta1]}, {t, 0, 20, 0.1}];
 Ylog2 = Table[
   {t, Limit[Ylog[t] /. alpha2 /.  xValues⟦i⟧ /. pi01 /. pi12 /. μf → 0.7 /.
      Finf → 2, beta1]}, {t, 0, 20, 0.1}];
 Ylog3 = Table[
   {t, Ylog[t] /. alpha1 /. beta2 /.  xValues⟦i⟧ /. pi02 /. pi11 /. μf → 0.7 /.
     Finf → 2}, {t, 0, 20, 0.1}];
 Ylog4 = Table[
   {t, Ylog[t] /. alpha2 /. beta2 /.  xValues⟦i⟧ /. pi02 /. pi12 /. μf → 0.7 /.
     Finf → 2}, {t, 0, 20, 0.1}];
 (*Prepare the data table*)
 YlogTable =
  Transpose[{Ylog1⟦All, 1⟧, Ylog1⟦All, 2⟧,
    Ylog2⟦All, 2⟧, Ylog3⟦All, 2⟧, Ylog4⟦All, 2⟧}];
 (*Export the data to a CSV file*)
 Export[
  datadir <> "Ylog_" <> fileNames⟦i⟧ <> "_combined.csv", YlogTable, "CSV"];
 , {i, 1, Length[xValues]}]
```

In[ ]:=
```
(*Iterate through the x0 values and
 export the yield data for exponential feed*)
Do[(*Calculate the data for the current x0 value*)
 Yexp1 =
  Table[{t, Limit[Yexp[t] /. alpha1 /. xValues⟦i⟧ /. pi01 /. pi11 /. μf → 0.7,
     beta1]}, {t, 0, 20, 0.1}];
 Yexp2 =
  Table[{t, Limit[Yexp[t] /. alpha2 /.  xValues⟦i⟧ /. pi01 /. pi12 /. μf → 0.7,
     beta1]}, {t, 0, 20, 0.1}];
 Yexp3 = Table[
   {t, Yexp[t] /. alpha1 /. beta2 /.  xValues⟦i⟧ /. pi02 /. pi11 /. μf → 0.7},
   {t, 0, 20, 0.1}];
 Yexp4 = Table[
   {t, Yexp[t] /. alpha2 /. beta2 /.  xValues⟦i⟧ /. pi02 /. pi12 /. μf → 0.7},
   {t, 0, 20, 0.1}];
 (*Prepare the data table*)
 YexpTable =
  Transpose[{Yexp1⟦All, 1⟧, Yexp1⟦All, 2⟧,
    Yexp2⟦All, 2⟧, Yexp3⟦All, 2⟧, Yexp4⟦All, 2⟧}];
 (*Export the data to a CSV file*)
 Export[
  datadir <> "Yexp_" <> fileNames⟦i⟧ <> "_combined.csv", YexpTable, "CSV"];
 , {i, 1, Length[xValues]}]
```

In[ ]:=
```
(*Iterate through the x0 values and
 export the yield data for constant feed*)
Do[(*Calculate the data for the current x0 value*)
 Ylin1 =
  Table[{t, Limit[Ylin[t] /. alpha1 /. xValues⟦i⟧ /. pi01 /. pi11, beta1]},
   {t, 0, 20, 0.1}];
 Ylin2 =
  Table[{t, Limit[Ylin[t] /. alpha2 /.  xValues⟦i⟧ /. pi01 /. pi12, beta1]},
   {t, 0, 20, 0.1}];
 Ylin3 = Table[{t, Ylin[t] /. alpha1 /. beta2 /.  xValues⟦i⟧ /. pi02 /. pi11},
   {t, 0, 20, 0.1}];
 Ylin4 = Table[{t, Ylin[t] /. alpha2 /. beta2 /.  xValues⟦i⟧ /. pi02 /. pi12},
   {t, 0, 20, 0.1}];
 (*Prepare the data table*)
 (*Prepare the data table*)
 YlinTable =
  Transpose[{Ylin1⟦All, 1⟧, Ylin1⟦All, 2⟧,
    Ylin2⟦All, 2⟧, Ylin3⟦All, 2⟧, Ylin4⟦All, 2⟧}];
 (*Export the data to a CSV file*)
 Export[
  datadir <> "Ylin_" <> fileNames⟦i⟧ <> "_combined.csv", YlinTable, "CSV"];
 , {i, 1, Length[xValues]}]
```

Define numeric yield functions for calculating yield across a range of production rates

In[ ]:=
```
YlinNum[t_, pi0_, pi1_] =
  Ylin[t] /. α → (1 / (1 + pi1 Yxs / Yps)) /. β → (pi0 Yxs / Yps + ρ Yxs / Yas) /.
        ρ → 0 /. Yieldxs /. Yieldps /. Yieldas /. π0 → pi0 /. π1 → pi1;
YexpNum[t_, pi0_, pi1_] =
  Yexp[t] /. α → (1 / (1 + pi1 Yxs / Yps)) /. β → (pi0 Yxs / Yps + ρ Yxs / Yas) /. μf →
        0.7 /. ρ → 0 /. Yieldxs /. Yieldps /. Yieldas /. π0 → pi0 /. π1 → pi1;
```

Loop over x0 values and calculate the maximum yield (+ time point) across production rate range for constant feed

```
Do[Module[{sol, maxValue, tValue, LinYieldMax, LinYieldMaxFlat},
  xValue = xValues⟦i⟧;

  LinYieldMax = Table[
    Module[{sol, maxValue, tValue},

     sol = FindMaximum[{YlinNum[t, pi0, pi1] /. xValue, 0 < t < 20}, {t, 0}];

     tValue = t /. sol⟦2⟧;

     maxValue = YlinNum[tValue, pi0, pi1] /. xValue;

     {pi0, pi1, tValue, maxValue}], {pi0, 0.1, 100, 1}, {pi1, 0.1, 100, 0.3}];
  LinYieldMaxFlat = Flatten[LinYieldMax, 1];

  Export[datadir <> "LinYieldMax_" <> fileNames⟦i⟧ <> ".csv",
   LinYieldMaxFlat, "CSV"];],
     {i, 1, Length[xValues]}]
```

Loop over x0 values and calculate the maximum yield (+ time point) across production rate range for exponential feed

```
Do[Module[{sol, maxValue, tValue, ExpYieldMax, ExpYieldMaxFlat},
  xValue = xValues⟦i⟧;

  ExpYieldMax = Table[
    Module[{sol, maxValue, tValue},

     sol = FindMaximum[{YexpNum[t, pi0, pi1] /. xValue, 0 < t < 20}, {t, 1}];

     tValue = t /. sol⟦2⟧;

     maxValue = YexpNum[tValue, pi0, pi1] /. xValue;

     {pi0, pi1, tValue, maxValue}], {pi0, 0.1, 100, 1}, {pi1, 0.1, 100, 0.3}];
  ExpYieldMaxFlat = Flatten[ExpYieldMax, 1];

  Export[datadir <> "ExpYieldMax_" <> fileNames⟦i⟧ <> ".csv",
   ExpYieldMaxFlat, "CSV"];],
     {i, 1, Length[xValues]}]
```

Define productivity functions for all feed rates , and the slope of productivity at t=0 for exponential and constant feed

```
In[•]:= LogProd[t_] = p[t] / t;
ExpProd[t_] = FullSimplify[pexp[t] / t];
LinProd[t_] = FullSimplify[plin[t] / t];
Dlinprodlim[t_] = Limit[D[LinProd[t], t], t → 0, Direction → -1];
Dexpprodlim[t_] = Limit[D[ExpProd[t], t], t → 0, Direction → -1];
```

Define numeric function for productivity for calculating the productivity across production rates

```
LinProdNum[t_, pi0_, pi1_] =
  LinProd[t] /. α → (1 / (1 + pi1 Yxs / Yps)) /. β → (pi0 Yxs / Yps + ρ Yxs / Yas) /.
        ρ → 0 /. Yieldxs /. Yieldps /. Yieldas /. π0 → pi0 /. π1 → pi1;
ExpProdNum[t_, pi0_, pi1_] =
  ExpProd[t] /. α → (1 / (1 + pi1 Yxs / Yps)) /. β → (pi0 Yxs / Yps + ρ Yxs / Yas) /.
         μf → 0.7 /. ρ → 0 /. Yieldxs /.
      Yieldps /. Yieldas /. π0 → pi0 /. π1 → pi1;
```

For constant and exponential feed: calculate the slope of productivity across production rates, keep only rates with ascending productivity and calculate their maximum. Then combine all into one table, use associations to ensure correct matching of production rates.

```
(* Helper function to process the x0 value,
calculate maximum productivities, combine and export them *)
processAndExport[xValue_, fileName_] :=
 Module[{DlinprodLim, flattenedDlinprodLim, DlinprodLimAscending,
   LinProdMax, DexpprodLim, flattenedDexpprodLim, DexpprodLimAscending,
   ExpProdMax, sol, maxValue, pi0, pi1, tValue, assocLinProdMax,
   assocExpProdMax, combinedKeys, combined, resultList, AllProd},
  Clear[DlinprodLim, flattenedDlinprodLim, DlinprodLimAscending,
   LinProdMax, DexpprodLim, flattenedDexpprodLim, DexpprodLimAscending,
   ExpProdMax, sol, maxValue, pi0, pi1, tValue, assocLinProdMax,
   assocExpProdMax, combinedKeys, combined, resultList, AllProd];

  DlinprodLim = Table[
    {pi0, pi1, Dlinprodlim[t] /. α → (1 / (1 + pi1 Yxs / Yps)) /. β → (pi0 Yxs / Yps +
               ρ Yxs / Yas) /. Yieldxs /. Yieldps /. Yieldas /. xValue /.
        ρ → 0 /. π0 → pi0 /. π1 → pi1}, {pi0, 0, 100, 1}, {pi1, 0, 100, 1}];
  flattenedDlinprodLim = Flatten[DlinprodLim, 1];
  Export[datadir <> "DlinprodLim_" <> fileName <> ".csv",
   flattenedDlinprodLim, "CSV"];
  DlinprodLimAscending = Select[flattenedDlinprodLim, #[[3]] > 0 &];

  LinProdMax = Table[
    Module[{sol, maxValue, pi0, pi1}, pi0 = DlinprodLimAscending[[row, 1]];
     pi1 = DlinprodLimAscending[[row, 2]];
     sol = Quiet[FindMaximum[{LinProdNum[t, pi0, pi1] /. xValue, 0 < t < 20},
        {t, 1}, MaxIterations → 1000], FindMaximum::cvmit];
     tValue = t /. sol[[2]];
```

```
    maxValue = LinProdNum[tValue, pi0, pi1] /. xValue;
    {pi0, pi1, tValue, maxValue}],
   {row, 1, Length[DlinprodLimAscending]}];
Export[datadir <> "LinProdMax_" <> fileName <> ".csv", LinProdMax, "CSV"];

DexpprodLim =
  Table[{pi0, pi1, Dexpprodlim[t] /. α → (1 / (1 + pi1 Yxs / Yps)) /. β →
              (pi0 Yxs / Yps + ρ Yxs / Yas) /. μf → 0.7 /. ρ → 0 /.
          Yieldxs /. Yieldps /. Yieldas /. xValue /. π0 → pi0 /.
     π1 → pi1}, {pi0, 0, 100, 1}, {pi1, 0, 100, 1}];
flattenedDexpprodLim = Flatten[DexpprodLim, 1];
Export[datadir <> "DexpprodLim_" <> fileName <> ".csv",
  flattenedDexpprodLim, "CSV"];
DexpprodLimAscending = Select[flattenedDexpprodLim, #[[3]] > 0 &];

ExpProdMax = Table[
   Module[{sol, maxValue, pi0, pi1}, pi0 = DexpprodLimAscending[[row, 1]];
    pi1 = DexpprodLimAscending[[row, 2]];
    sol = Quiet[FindMaximum[{ExpProdNum[t, pi0, pi1] /. xValue, 0 < t < 20},
       {t, 1}, MaxIterations → 1000], FindMaximum::cvmit];
    tValue = t /. sol[[2]];
    maxValue = ExpProdNum[tValue, pi0, pi1] /. xValue;
    {pi0, pi1, tValue, maxValue}],
   {row, 1, Length[DexpprodLimAscending]}];
Export[datadir <> "ExpProdMax_" <> fileName <> ".csv", ExpProdMax, "CSV"];

assocLinProdMax = Association[
   Map[Function[{p}, Rule[{p[[1]], p[[2]]}, {p[[3]], p[[4]]}]], LinProdMax]];
assocExpProdMax = Association[
   Map[Function[{p}, Rule[{p[[1]], p[[2]]}, {p[[3]], p[[4]]}]], ExpProdMax]];

combinedKeys = Union[Keys[assocLinProdMax], Keys[assocExpProdMax]];

combined = Association[Map[Function[key, key → {
       If[KeyExistsQ[assocLinProdMax, key],
        First[assocLinProdMax[key]], Missing[]],
       If[KeyExistsQ[assocLinProdMax, key],
        Last[assocLinProdMax[key]], Missing[]],
       If[KeyExistsQ[assocExpProdMax, key],
        First[assocExpProdMax[key]], Missing[]],
       If[KeyExistsQ[assocExpProdMax, key],
        Last[assocExpProdMax[key]], Missing[]]}], combinedKeys]];

resultList = Normal[combined];

AllProd =
  Map[{#[[1, 1]], #[[1, 2]], #[[2, 1]], #[[2, 2]], #[[2, 3]], #[[2, 4]]} &, resultList];
```

```
  Export[datadir <> "AllProd_" <> fileName <> ".csv", AllProd, "CSV"];]

(*Main loop to process all x0 values with corresponding fileNames*)
Do[processAndExport[xValues⟦index⟧, fileNames⟦index⟧],
 {index, 1, Length[xValues]}]
```

Export productivity data for distinct production rates across all x0 values and feed strategies.

```
(*Iterate through the x0 values and
 export productivity data for logistic feed *)
Do[(*Calculate the data for the current x0 value*)
  prodlog1 =
   Table[{t, Quiet[Limit[LogProd[t] /. alpha1 /. pi01 /. pi11 /. xValues⟦i⟧ /.
         μf → 0.7 /. Finf → 2, beta1]]}, {t, 0, 20, 0.1}];
  prodlog2 = Table[
    {t, Quiet[Limit[LogProd[t] /. alpha2 /. pi01 /. pi12 /.  xValues⟦i⟧ /.
         μf → 0.7 /. Finf → 2, beta1]]}, {t, 0, 20, 0.1}];
  prodlog3 = Table[
    {t, Quiet[LogProd[t] /. alpha1 /. beta2 /. pi02 /. pi11 /.  xValues⟦i⟧ /.
        μf → 0.7 /. Finf → 2]}, {t, 0, 20, 0.1}];
  prodlog4 = Table[
    {t, Quiet[LogProd[t] /. alpha2 /. beta2 /. pi02 /. pi12 /.  xValues⟦i⟧ /.
        μf → 0.7 /. Finf → 2]}, {t, 0, 20, 0.1}];
  (*Prepare the data table*)
  prodlogTable =
   Transpose[{prodlog1⟦All, 1⟧, prodlog1⟦All, 2⟧,
     prodlog2⟦All, 2⟧, prodlog3⟦All, 2⟧, prodlog4⟦All, 2⟧}];
  (*Export the data to a CSV file*)
  Export[datadir <> "prodlog_" <> fileNames⟦i⟧ <> "_combined.csv",
   prodlogTable, "CSV"];
  , {i, 1, Length[xValues]}];
```

```
(*Iterate through the x0 values and export
 the productivity data for exponential feed*)
Do[(*Calculate the data for the current x0 value*)
 prodexp1 = Table[
   {t, Quiet[ExpProd[t] /. alpha1 /. beta1 /. pi01 /. pi11 /. xValues⟦i⟧ /.
      μf → 0.7]}, {t, 0, 20, 0.1}];
 prodexp2 =
  Table[{t, Quiet[Limit[ExpProd[t] /. alpha2 /. pi01 /. pi12 /.  xValues⟦i⟧ /.
       μf → 0.7, beta1]]}, {t, 0, 20, 0.1}];
 prodexp3 = Table[
   {t, Quiet[ExpProd[t] /. alpha1 /. beta2 /. pi02 /. pi11 /.  xValues⟦i⟧ /.
      μf → 0.7]}, {t, 0, 20, 0.1}];
 prodexp4 = Table[
   {t, Quiet[ExpProd[t] /. alpha2 /. beta2 /. pi02 /. pi12 /.  xValues⟦i⟧ /.
      μf → 0.7]}, {t, 0, 20, 0.1}];
 (*Prepare the data table*)
 prodexpTable =
  Transpose[{prodexp1⟦All, 1⟧, prodexp1⟦All, 2⟧,
    prodexp2⟦All, 2⟧, prodexp3⟦All, 2⟧, prodexp4⟦All, 2⟧}];
 (*Export the data to a CSV file*)
 Export[datadir <> "prodexp_" <> fileNames⟦i⟧ <> "_combined.csv",
  prodexpTable, "CSV"];
 , {i, 1, Length[xValues]}]
```

```
(*Iterate through the x0 values and
 export the productivity data for constant feed *)
Do[(*Calculate the data for the current x0 value*)
 prodlin1 = Table[
   {t, Quiet[LinProd[t] /. alpha1 /. beta1 /. pi01 /. pi11 /. xValues⟦i⟧]},
   {t, 0, 20, 0.1}];
 prodlin2 =
  Table[{t, Quiet[Limit[LinProd[t] /. alpha2 /. pi01 /. pi12 /.  xValues⟦i⟧,
      beta1]]}, {t, 0, 20, 0.1}];
 prodlin3 = Table[
   {t, Quiet[LinProd[t] /. alpha1 /. beta2 /. pi02 /. pi11 /.  xValues⟦i⟧]},
   {t, 0, 20, 0.1}];
 prodlin4 = Table[
   {t, Quiet[LinProd[t] /. alpha2 /. beta2 /. pi02 /. pi12 /.  xValues⟦i⟧]},
   {t, 0, 20, 0.1}];
 (*Prepare the data table*)
 prodlinTable =
  Transpose[{prodlin1⟦All, 1⟧, prodlin1⟦All, 2⟧,
    prodlin2⟦All, 2⟧, prodlin3⟦All, 2⟧, prodlin4⟦All, 2⟧}];
 (*Export the data to a CSV file*)
 Export[datadir <> "prodlin_" <> fileNames⟦i⟧ <> "_combined.csv",
  prodlinTable, "CSV"];
 , {i, 1, Length[xValues]}]
```