```
> #Maple code for example 2 of paper Parameter Redundancy and Identifiability in Hidden Markov
      Models by D.J. Cole
> with(LinearAlgebra) :
> Dmat :=proc(se, pars)
  local DD1, i, j;
  description "This procedure finds the derivative matrix for vector se with parameters pars";
  with(LinearAlgebra) :
  DD1 := Matrix(1 ..Dimension(pars), 1 ..Dimension(se)) :
  for i from 1 to Dimension(pars)  do
      for j from 1 to Dimension(se) do
          DD1[i, j] := diff(se[j], pars[i])
      end do
   end do;
  DD1;
   end proc:
> Estpar :=proc(DD1, pars, ret)
  local r, d, alphapre, alpha, PDE, FF, i, ans;
  description "Finds the estimable set of parameters for derivative matrix DD1. If ret = 1 returns
      alpha, PDEs, estimable parameter combinations. Otherwise returns estimable parameter
      combinations";
  with(LinearAlgebra) :
  r := Rank(DD1); d := Dimension(pars) - r :
  alphapre := NullSpace(Transpose(DD1)) : α := Matrix(d, Dimension(pars)) : PDE :=
       Vector(d) :
  FF := f(seq(pars[i], i = 1 ..Dimension(pars))) :
  for i from 1 to d do
         α[i, 1 ..Dimension(pars)] := alphapre[i] :
         PDE[i] := add(diff(FF, pars[j])·α[i, j], j = 1 ..Dimension(pars)) :
  end do:
   if ret = 1 then
           ans := ⟨pdsolve({seq(PDE[i] = 0, i = 1 ..d)}), {alpha}, {PDE}⟩ :
   else
          ans := pdsolve({seq(PDE[i] = 0, i = 1 ..d)}) :
   end if:
   ans :
  end proc:
>
> # Symbolic method that uses probabilities of each history as exhaustive summary
> # 3 years of data
> #List of possible capture histories
> H := ⟨⟨1|2|0⟩, ⟨1|1|2⟩, ⟨1|1|1⟩, ⟨1|1|0⟩, ⟨1|0|2⟩, ⟨1|0|1⟩, ⟨1|0|0⟩, ⟨0|1|2⟩, ⟨0|1|1⟩, ⟨0|1|0⟩⟩
```

(1)

$$H := \begin{bmatrix} 1 & 2 & 0 \\ 1 & 1 & 2 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 2 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 2 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{1}$$

```
> # Using HMM form to create the probabilities of each history
> m := 3 :
> deltavector := <1 |0|0> :
> Py0 := DiagonalMatrix(<1 - p[t], 1 - lambda[t], 1>);
```

$$Py0 := \begin{bmatrix} 1 - p_t & 0 & 0 \\ 0 & 1 - \lambda_t & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

```
> Py1 := DiagonalMatrix(<p[t], 0, 0>);
```

$$Py1 := \begin{bmatrix} p_t & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3}$$

```
> Py2 := DiagonalMatrix(<0, lambda[t], 0>);
```

$$Py2 := \begin{bmatrix} 0 & 0 & 0 \\ 0 & \lambda_t & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4}$$

```
>
> Gammamatrix := <<phi[t]|1 - phi[t]|0>, <0|0|1>, <0|0|1>>;
```

$$Gammamatrix := \begin{bmatrix} \phi_t & 1 - \phi_t & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

```
> # The exhuastive summary consisting of the probabilties of each history:
> kappa := Vector(10) :
> kappa[1] := deltavector · eval(Gammamatrix, t = 1).eval(Py2, t = 2).eval(Gammamatrix, t = 2)
      .eval(Py0, t = 3).Vector(m, 1, 1) :
> kappa[2] := deltavector · eval(Gammamatrix, t = 1).eval(Py1, t = 2).eval(Gammamatrix, t = 2)
```

```
        .eval(Py2, t=3).Vector(m, 1, 1) :
> kappa[3] := deltavector·eval(Gammamatrix, t=1).eval(Py1, t=2).eval(Gammamatrix, t=2)
        .eval(Py1, t=3).Vector(m, 1, 1) :
> kappa[4] := deltavector·eval(Gammamatrix, t=1).eval(Py1, t=2).eval(Gammamatrix, t=2)
        .eval(Py0, t=3).Vector(m, 1, 1) :
> kappa[5] := deltavector·eval(Gammamatrix, t=1).eval(Py0, t=2).eval(Gammamatrix, t=2)
        .eval(Py2, t=3).Vector(m, 1, 1) :
> kappa[6] := deltavector·eval(Gammamatrix, t=1).eval(Py0, t=2).eval(Gammamatrix, t=2)
        .eval(Py1, t=3).Vector(m, 1, 1) :
> kappa[7] := deltavector·eval(Gammamatrix, t=1).eval(Py0, t=2).eval(Gammamatrix, t=2)
        .eval(Py0, t=3).Vector(m, 1, 1) :
> kappa[8] := deltavector·eval(Gammamatrix, t=2).eval(Py2, t=3).Vector(m, 1, 1) :
> kappa[9] := deltavector·eval(Gammamatrix, t=2).eval(Py1, t=3).Vector(m, 1, 1) :
> kappa[10] := deltavector·eval(Gammamatrix, t=2).eval(Py0, t=3).Vector(m, 1, 1) :
> kappa
```

$$\begin{bmatrix} (1-\phi_1)\lambda_2 \\ \phi_1 p_2 (1-\phi_2)\lambda_3 \\ \phi_1 p_2 \phi_2 p_3 \\ \phi_1 p_2 \phi_2 (1-p_3) + \phi_1 p_2 (1-\phi_2)(1-\lambda_3) \\ \phi_1 (1-p_2)(1-\phi_2)\lambda_3 \\ \phi_1 (1-p_2)\phi_2 p_3 \\ \phi_1 (1-p_2)\phi_2(1-p_3) + \phi_1(1-p_2)(1-\phi_2)(1-\lambda_3) + (1-\phi_1)(1-\lambda_2) \\ (1-\phi_2)\lambda_3 \\ \phi_2 p_3 \\ \phi_2 (1-p_3) + (1-\phi_2)(1-\lambda_3) \end{bmatrix} \quad \textbf{(6)}$$

```
> # Vector of parameters:
> pars := ⟨φ1, φ2, p2, p3, λ2, λ3⟩ :
> D1 := simplify(Dmat(kappa, pars));
```

$$D1 := [[-\lambda_2, -p_2(-1+\phi_2)\lambda_3, p_2\phi_2p_3, -((p_3-\lambda_3)\phi_2+\lambda_3-1)p_2, (-1+p_2)(-1+\phi_2)\lambda_3, -(-1+p_2)\phi_2p_3, -(-1+p_2)(-1+\phi_2)\lambda_3+(-1+p_2)\phi_2p_3-p_2+\lambda_2, 0, 0, 0],$$
$$[0, -\phi_1p_2\lambda_3, \phi_1p_2p_3, \phi_1p_2(-p_3+\lambda_3), \phi_1(-1+p_2)\lambda_3, -\phi_1(-1+p_2)p_3, \phi_1(p_3-\lambda_3)(-1+p_2), -\lambda_3, p_3, -p_3+\lambda_3],$$
$$[0, -\phi_1(-1+\phi_2)\lambda_3, \phi_1\phi_2p_3, -((p_3-\lambda_3)\phi_2+\lambda_3-1)\phi_1, \phi_1(-1+\phi_2)\lambda_3, \quad \textbf{(7)}$$

$-\phi_1 \phi_2 p_3, \left(\left(1-\phi_2\right)\lambda_3+\phi_2 p_3-1\right)\phi_1, 0, 0, 0\Big],$

$\Big[0, 0, \phi_1 p_2 \phi_2, -\phi_1 p_2 \phi_2, 0, -\phi_1 (-1+p_2)\phi_2, \phi_1(-1+p_2)\phi_2, 0, \phi_2, -\phi_2\Big],$

$\Big[1-\phi_1, 0, 0, 0, 0, 0, -1+\phi_1, 0, 0, 0\Big],$

$\Big[0, -\phi_1 p_2 \left(-1+\phi_2\right), 0, \phi_1 p_2\left(-1+\phi_2\right), \phi_1 (-1+p_2)\left(-1+\phi_2\right), 0, -\phi_1(-1+p_2)\Big($

$-1+\phi_2\Big), 1-\phi_2, 0, -1+\phi_2\Big]\Big]$

```
> r := Rank(D1); d := Dimension(pars) - r;
```

$$r := 5$$
$$d := 1 \tag{8}$$

```
> simplify(Estpar(D1, pars, 1));
```

$$\left[\left[\left\{f\left(\phi_1, \phi_2, p_2, p_3, \lambda_2, \lambda_3\right) = \_F1\left(\phi_1, p_2, \lambda_2, \lambda_3\left(-1+\phi_2\right), \phi_2 p_3\right)\right\}\right], \tag{9}\right.$$

$$\left[\left\{\left[\begin{matrix} 0 & \frac{1-\phi_2}{\lambda_3} & 0 & \frac{\left(-1+\phi_2\right)p_3}{\lambda_3 \phi_2} & 0 & 1\end{matrix}\right]\right\}\right],$$

$$\left[\left\{\left[-\frac{\left(\frac{\partial}{\partial \phi_2} f\left(\phi_1, \phi_2, p_2, p_3, \lambda_2, \lambda_3\right)\right)\left(-1+\phi_2\right)}{\lambda_3} + \frac{\left(\frac{\partial}{\partial p_3} f\left(\phi_1, \phi_2, p_2, p_3, \lambda_2, \lambda_3\right)\right)\left(-1+\phi_2\right)p_3}{\lambda_3 \phi_2} + \frac{\partial}{\partial \lambda_3} f\left(\phi_1, \phi_2, p_2, p_3, \lambda_2, \lambda_3\right)\right]\right\}\right]\Bigg]$$

```
>
> # Symbolic method using HMM exhaustive summary:
> kappa := Vector(12) :
> # History 120
  kappa[1] :=  deltavector · eval(Gammamatrix, t = 1).eval(Py2, t = 2).Vector(m, 1, 1) :
   # Identical term so not needed: deltavector.eval(Gammamatrix, t = 1).eval(Py2, t = 2)
      .eval(Gammamatrix, t = 2).eval(Py0, t = 3).Vector(m, 1, 1) :
> # History 112
  kappa[2] := deltavector · eval(Gammamatrix, t = 1).eval(Py1, t = 2).Vector(m, 1, 1) :
  kappa[3] := deltavector · eval(Gammamatrix, t = 1).eval(Py1, t = 2).eval(Gammamatrix, t = 2)
      .eval(Py2, t = 3).Vector(m, 1, 1) :
> #History 111
   # Identical term so not needed:  deltavector.eval(Gammamatrix, t = 1).eval(Py1, t = 2)
      .Vector(m, 1, 1) :
  kappa[4] := deltavector · eval(Gammamatrix, t = 1).eval(Py1, t = 2).eval(Gammamatrix, t = 2)
      .eval(Py1, t = 3).Vector(m, 1, 1) :
> #History 110
```

```
# Identical term so not needed: deltavector.eval(Gammamatrix, t = 1).eval(Py1, t = 2).Vector(m,
    1, 1) :
kappa[5] := deltavector · eval(Gammamatrix, t = 1).eval(Py1, t = 2).eval(Gammamatrix, t = 2)
    .eval(Py0, t = 3).Vector(m, 1, 1) :
> #History 102
kappa[6] := deltavector · eval(Gammamatrix, t = 1).eval(Py0, t = 2).Vector(m, 1, 1) :
 kappa[7] := deltavector · eval(Gammamatrix, t = 1).eval(Py0, t = 2).eval(Gammamatrix, t = 2)
    .eval(Py2, t = 3).Vector(m, 1, 1) :
> #History 101
 # Identical term so not needed:  deltavector.eval(Gammamatrix, t = 1).eval(Py0, t = 2)
    .Vector(m, 1, 1) :
kappa[8] := deltavector · eval(Gammamatrix, t = 1).eval(Py0, t = 2).eval(Gammamatrix, t = 2)
    .eval(Py1, t = 3).Vector(m, 1, 1) :
> # History 100
# Identical term so not needed:  deltavector.eval(Gammamatrix, t = 1).eval(Py0, t = 2)
    .Vector(m, 1, 1) :
kappa[9] := deltavector · eval(Gammamatrix, t = 1).eval(Py0, t = 2).eval(Gammamatrix, t = 2)
    .eval(Py0, t = 3).Vector(m, 1, 1) :
> # History 012
kappa[10] := deltavector · eval(Gammamatrix, t = 2).eval(Py2, t = 3).Vector(m, 1, 1) :
> # History 011
kappa[11] := deltavector · eval(Gammamatrix, t = 2).eval(Py1, t = 3).Vector(m, 1, 1) :
> # History 010
kappa[12] := deltavector · eval(Gammamatrix, t = 2).eval(Py0, t = 3).Vector(m, 1, 1) :
> kappa[1 ..10]; kappa[11 ..12]
```

$$\begin{bmatrix} (1-\phi_1)\lambda_2 \\ \phi_1 p_2 \\ \phi_1 p_2 (1-\phi_2)\lambda_3 \\ \phi_1 p_2 \phi_2 p_3 \\ \phi_1 p_2 \phi_2 (1-p_3) + \phi_1 p_2 (1-\phi_2)(1-\lambda_3) \\ \phi_1 (1-p_2) + (1-\phi_1)(1-\lambda_2) \\ \phi_1 (1-p_2)(1-\phi_2)\lambda_3 \\ \phi_1 (1-p_2)\phi_2 p_3 \\ \phi_1 (1-p_2)\phi_2(1-p_3) + \phi_1(1-p_2)(1-\phi_2)(1-\lambda_3) + (1-\phi_1)(1-\lambda_2) \\ (1-\phi_2)\lambda_3 \end{bmatrix}$$

$$\begin{bmatrix} \phi_2 p_3 \\ \phi_2(1-p_3) + (1-\phi_2)(1-\lambda_3) \end{bmatrix} \tag{10}$$

>

> *deltavector* • *eval*(*Gammamatrix*, $t=2$).*eval*(*Py0*, $t=3$).*eval*(*Gammamatrix*, $t=3$).*eval*(*Py1*, $t=4$).*eval*(*Gammamatrix*, $t=4$).*eval*(*Py2*, $t=5$).*Vector*(m, 1, 1);

$$\phi_2 \left(1-p_3\right) \phi_3 p_4 \left(1-\phi_4\right) \lambda_5 \tag{11}$$

> *pars* := $\langle \phi_1, \phi_2, p_2, p_3, \lambda_2, \lambda_3 \rangle$:

> *D1* := *Dmat*(kappa, *pars*) :

> r := *Rank*(*D1*); d := *Dimension*(*pars*) − r;

$$r := 5$$
$$d := 1 \tag{12}$$

> *simplify*(*Estpar*(*D1*, *pars*, 1));

$$\left[\left[\left\{f\left(\phi_1, \phi_2, p_2, p_3, \lambda_2, \lambda_3\right) = \_F1\left(\phi_1, p_2, \lambda_2, \left(-1+\phi_2\right)\lambda_3, \phi_2 p_3\right)\right\}\right], \tag{13}$$

$$\left[\left\{\left[\begin{array}{cccccc} 0 & \frac{1-\phi_2}{\lambda_3} & 0 & \frac{\left(-1+\phi_2\right) p_3}{\lambda_3 \phi_2} & 0 & 1 \end{array}\right]\right\}\right],$$

$$\left[\left\{\left[-\frac{\left(\frac{\partial}{\partial \phi_2} f\left(\phi_1, \phi_2, p_2, p_3, \lambda_2, \lambda_3\right)\right)\left(-1+\phi_2\right)}{\lambda_3} + \frac{\left(\frac{\partial}{\partial p_3} f\left(\phi_1, \phi_2, p_2, p_3, \lambda_2, \lambda_3\right)\right)\left(-1+\phi_2\right) p_3}{\lambda_3 \phi_2} + \frac{\partial}{\partial \lambda_3} f\left(\phi_1, \phi_2, p_2, p_3, \lambda_2, \lambda_3\right)\right]\right\}\right]\right]$$

> *# Adding an extra year of data, histories that add new terms not already in the above exhaustive summary:*

> *# Note 1200, 1120, etc result in terms that are identical to terms that are already in the above exhaustive summary*

> *kappaex* := *Vector*(21) :

> *# 1112*
kappaex[1] := *deltavector* • *eval*(*Gammamatrix*, $t=1$).*eval*(*Py1*, $t=1$).*eval*(*Gammamatrix*, $t=2$).*eval*(*Py1*, $t=3$).*eval*(*Gammamatrix*, $t=3$).*eval*(*Py2*, $t=4$).*Vector*(m, 1, 1);

$$kappaex_1 := \phi_1 p_1 \phi_2 p_3 \left(1-\phi_3\right) \lambda_4 \tag{14}$$

> *# 1111*
kappaex[2] := *deltavector* • *eval*(*Gammamatrix*, $t=1$).*eval*(*Py1*, $t=1$).*eval*(*Gammamatrix*, $t=2$).*eval*(*Py1*, $t=3$).*eval*(*Gammamatrix*, $t=3$).*eval*(*Py1*, $t=4$).*Vector*(m, 1, 1);

$$kappaex_2 := \phi_1 p_1 \phi_2 p_3 \phi_3 p_4 \tag{15}$$

> *# 1110*
kappaex[3] := *deltavector* • *eval*(*Gammamatrix*, $t=1$).*eval*(*Py1*, $t=1$).*eval*(*Gammamatrix*, $t=2$).*eval*(*Py1*, $t=3$).*eval*(*Gammamatrix*, $t=3$).*eval*(*Py0*, $t=4$).*Vector*(m, 1, 1);

$$kappaex_3 := \phi_1 p_1 \phi_2 p_3 \phi_3 \left(1-p_4\right) + \phi_1 p_1 \phi_2 p_3 \left(1-\phi_3\right)\left(1-\lambda_4\right) \tag{16}$$

> # 1102

kappaex[4] := deltavector · eval(Gammamatrix, t = 1).eval(Py1, t = 1).eval(Gammamatrix, t = 2).eval(Py0, t = 3).eval(Gammamatrix, t = 3).eval(Py2, t = 4).Vector(m, 1, 1);

$$kappaex_4 := \phi_1 p_1 \phi_2 (1 - p_3) (1 - \phi_3) \lambda_4 \tag{17}$$

> # 1101

kappaex[5] := deltavector · eval(Gammamatrix, t = 1).eval(Py1, t = 1).eval(Gammamatrix, t = 2).eval(Py0, t = 3).eval(Gammamatrix, t = 3).eval(Py1, t = 4).Vector(m, 1, 1);

$$kappaex_5 := \phi_1 p_1 \phi_2 (1 - p_3) \phi_3 p_4 \tag{18}$$

> # 1100

kappaex[6] := deltavector · eval(Gammamatrix, t = 1).eval(Py1, t = 1).eval(Gammamatrix, t = 2).eval(Py0, t = 3).eval(Gammamatrix, t = 3).eval(Py0, t = 4).Vector(m, 1, 1);

$$kappaex_6 := \phi_1 p_1 \phi_2 (1 - p_3) \phi_3 (1 - p_4) + \phi_1 p_1 \phi_2 (1 - p_3) (1 - \phi_3) (1 - \lambda_4) + \phi_1 p_1 (1 - \phi_2) (1 - \lambda_3) \tag{19}$$

> # 1012

kappaex[7] := deltavector · eval(Gammamatrix, t = 1).eval(Py0, t = 1).eval(Gammamatrix, t = 2).eval(Py1, t = 3).eval(Gammamatrix, t = 3).eval(Py2, t = 4).Vector(m, 1, 1);

$$kappaex_7 := \phi_1 (1 - p_1) \phi_2 p_3 (1 - \phi_3) \lambda_4 \tag{20}$$

> # 1011

kappaex[8] := deltavector · eval(Gammamatrix, t = 1).eval(Py0, t = 1).eval(Gammamatrix, t = 2).eval(Py1, t = 3).eval(Gammamatrix, t = 3).eval(Py1, t = 4).Vector(m, 1, 1);

$$kappaex_8 := \phi_1 (1 - p_1) \phi_2 p_3 \phi_3 p_4 \tag{21}$$

> # 1010

kappaex[9] := deltavector · eval(Gammamatrix, t = 1).eval(Py0, t = 1).eval(Gammamatrix, t = 2).eval(Py1, t = 3).eval(Gammamatrix, t = 3).eval(Py0, t = 4).Vector(m, 1, 1);

$$kappaex_9 := \phi_1 (1 - p_1) \phi_2 p_3 \phi_3 (1 - p_4) + \phi_1 (1 - p_1) \phi_2 p_3 (1 - \phi_3) (1 - \lambda_4) \tag{22}$$

> # 1002

kappaex[10] := deltavector · eval(Gammamatrix, t = 1).eval(Py0, t = 1).eval(Gammamatrix, t = 2).eval(Py0, t = 3).eval(Gammamatrix, t = 3).eval(Py2, t = 4).Vector(m, 1, 1);

$$kappaex_{10} := \phi_1 (1 - p_1) \phi_2 (1 - p_3) (1 - \phi_3) \lambda_4 \tag{23}$$

> # 1001

kappaex[11] := deltavector · eval(Gammamatrix, t = 1).eval(Py0, t = 1).eval(Gammamatrix, t = 2).eval(Py0, t = 3).eval(Gammamatrix, t = 3).eval(Py1, t = 4).Vector(m, 1, 1);

$$kappaex_{11} := \phi_1 (1 - p_1) \phi_2 (1 - p_3) \phi_3 p_4 \tag{24}$$

> # 1000

kappaex[12] := deltavector · eval(Gammamatrix, t = 1).eval(Py0, t = 1).eval(Gammamatrix, t = 2).eval(Py0, t = 3).eval(Gammamatrix, t = 3).eval(Py0, t = 4).Vector(m, 1, 1);

$$kappaex_{12} := \phi_1 (1 - p_1) \phi_2 (1 - p_3) \phi_3 (1 - p_4) + \phi_1 (1 - p_1) \phi_2 (1 - p_3) (1 - \phi_3) (1 - \lambda_4) + \phi_1 (1 - p_1) (1 - \phi_2) (1 - \lambda_3) + (1 - \phi_1) (1 - \lambda_1) \tag{25}$$

> # 0112

kappaex[13] := deltavector · eval(Gammamatrix, t = 2).eval(Py1, t = 3).eval(Gammamatrix, t = 3).eval(Py2, t = 4).Vector(m, 1, 1);

$$kappaex_{13} := \phi_2 p_3 (1-\phi_3) \lambda_4 \quad \textbf{(26)}$$

> # 0111
kappaex[14] := deltavector · eval(Gammamatrix, t = 2).eval(Py1, t = 3).eval(Gammamatrix, t = 3).eval(Py1, t = 4).Vector(m, 1, 1);

$$kappaex_{14} := \phi_2 p_3 \phi_3 p_4 \quad \textbf{(27)}$$

> # 0110
kappaex[15] := deltavector · eval(Gammamatrix, t = 2).eval(Py1, t = 3).eval(Gammamatrix, t = 3).eval(Py0, t = 4).Vector(m, 1, 1);

$$kappaex_{15} := \phi_2 p_3 \phi_3 (1-p_4) + \phi_2 p_3 (1-\phi_3)(1-\lambda_4) \quad \textbf{(28)}$$

> # 0102
kappaex[16] := deltavector · eval(Gammamatrix, t = 2).eval(Py0, t = 3).eval(Gammamatrix, t = 3).eval(Py2, t = 4).Vector(m, 1, 1);

$$kappaex_{16} := \phi_2 (1-p_3)(1-\phi_3) \lambda_4 \quad \textbf{(29)}$$

> # 0101
kappaex[17] := deltavector · eval(Gammamatrix, t = 2).eval(Py0, t = 3).eval(Gammamatrix, t = 3).eval(Py1, t = 4).Vector(m, 1, 1);

$$kappaex_{17} := \phi_2 (1-p_3) \phi_3 p_4 \quad \textbf{(30)}$$

> # 0100
kappaex[18] := deltavector · eval(Gammamatrix, t = 2).eval(Py0, t = 3).eval(Gammamatrix, t = 3).eval(Py0, t = 4).Vector(m, 1, 1);

$$kappaex_{18} := \phi_2 (1-p_3) \phi_3 (1-p_4) + \phi_2 (1-p_3)(1-\phi_3)(1-\lambda_4) + (1-\phi_2)(1-\lambda_3) \quad \textbf{(31)}$$

> # 0012
kappaex[19] := deltavector · eval(Gammamatrix, t = 3).eval(Py2, t = 4).Vector(m, 1, 1);

$$kappaex_{19} := (1-\phi_3) \lambda_4 \quad \textbf{(32)}$$

> # 0011
kappaex[20] := deltavector · eval(Gammamatrix, t = 3).eval(Py1, t = 4).Vector(m, 1, 1);

$$kappaex_{20} := \phi_3 p_4 \quad \textbf{(33)}$$

> # 0010
kappaex[21] := deltavector · eval(Gammamatrix, t = 3).eval(Py0, t = 4).Vector(m, 1, 1);

$$kappaex_{21} := \phi_3 (1-p_4) + (1-\phi_3)(1-\lambda_4) \quad \textbf{(34)}$$

> # Reparameterise kappaex in terms of the estimable parameter combination ϕ_1, p_2, λ_2, beta[1] $= (-1+\phi_2)\lambda_3$, beta[2] $= \phi_2 p_3$

> A := solve({beta[1] = $(-1+\phi_2)\lambda_3$, beta[2] = $\phi_2 p_3$}, {lambda[3], p[3]})

$$A := \left\{ \lambda_3 = \frac{\beta_1}{-1+\phi_2}, p_3 = \frac{\beta_2}{\phi_2} \right\} \quad \textbf{(35)}$$

> kappaexre := simplify(eval(kappaex, A));

$$kappaexre := \left[\begin{array}{c} \text{1 .. 21 } Vector_{column} \\ \text{Data Type: anything} \\ \text{Storage: rectangular} \\ \text{Order: Fortran\_order} \end{array}\right] \qquad \textbf{(36)}$$

> *indets*(*kappaexre*)

$$\{\beta_1, \beta_2, \lambda_1, \lambda_4, p_1, p_4, \phi_1, \phi_2, \phi_3\} \qquad \textbf{(37)}$$

> *# When we reparameterise the parameters are* $\phi_1, p_2, \lambda_2, \beta_1, \beta_2$, *so the extra parameters are*

> *pars2* := $\langle \phi_2, \phi_3, \lambda_4, p_4 \rangle$:

> *Dex* := *Dmat*(*kappaex*, *pars2*)

$$Dex := \left[\begin{array}{c} \text{4 x 21 Matrix} \\ \text{Data Type: anything} \\ \text{Storage: rectangular} \\ \text{Order: Fortran\_order} \end{array}\right] \qquad \textbf{(38)}$$

> *rex* := *Rank*(*Dex*)

$$rex := 3 \qquad \textbf{(39)}$$

> *Estpar*(*Dex*, *pars2*, 0)

$$\{f(\phi_2, \phi_3, \lambda_4, p_4) = \_F1(\phi_2, (-1 + \phi_3)\,\lambda_4, \phi_3\,p_4)\} \qquad \textbf{(40)}$$

> # Model rank = 5+3, but there are 9 parameters in the orgional model therefore it is parameter redundant

> *# The estimable parameter cominations are* ϕ_2, $(-1+\phi_3)\,\lambda_4$, $\phi_3\,p_4$,
as we can now estimate ϕ_2 *we can deduce that we can now also estimate* p_2 and λ_2

> # By the extension Theorem model will have rank 3 T−1, but there are 3 T parameters, so we will always be parameter redundant with deficency 1.

>