Model Code

Code for the Capital Fisher Model

See section 4.2 (p. 95) for details. The variables are transformed to replace $D_k v$ by `mvk` $:= -D_k v$ to meet technical limitations of the script language. Some cases where one variables obtains it upper or lower bound are excluded by the `unreachable-values` directive.

```
(quantity-spaces
 (x   (0 xmsy Q xmax)  "Stock")
 (k   (0 kmax)         "Capital")
 (I   (0 Imax)         "Investment")
 (R   (0 Rmsy Rmax)    "Recruitment")
 (h   (0 MSY hmax)     "Harvest")
 (mvk (0 inf)          "-D_k v")
 (dx  (minf 0 inf)     "dx")
 (dk  (minf 0 inf)     "dk")
 (dI  (minf 0 inf)     "dI"))

(constraints

 ;; derivatives
 ((d//dt x dx))
 ((d//dt k dk))
 ((d//dt I dI))

 ((add dk k I) (0 0 0))
 ((add dx h R) (0 0 0)  (0 MSY Rmsy))
 ((add dI mvk I) (0 0 0))

 ;; auxiliary variables
 ((U- x R) (xmsy Rmsy)  (0 0) (Q 0))
 (((M + - -) h x k mvk))

 ;; algebraic constraints
 (((M + +) k x h))

 ;; sigma (only second, unambiguous row)
  (((M - + ) k I dk)) ; implicit in (add dk k I)
```

```
 ;; exclusion of marginal edges
 ((cornot x h) (xmsy MSY))
 ((cornot x dx) (xmsy 0))
 ((cornot x dk) (xmsy 0))
 ((cornot x dI) (xmsy 0))
 ((cornot h dx) (MSY 0))
 ((cornot h dk) (MSY 0))
 ((cornot h dI) (MSY 0)))

;;; technical directives

(unreachable-values
  (x xmax) (k kmax) (I 0 Imax) (R Rmax) (h hmax) (mvk 0))
```

Code of the Negotiation Model with Capital

The basic model with unconstrained recommendations r and restricted quantitiy space (see section 4.3.2, p. 109).

```
(quantity-spaces
 (x (xmin xv xmsy)  "Stock")
 (k (0 kmax)        "Capital")
 (h (hmin MSY hmax) "Harvest")
 (r (0 hmin rmax)   "Catch Recommendation")
 (dx (minf 0 inf)   "dx")
 (dk (minf 0 inf)   "dk")
 (dh (minf 0 inf)   "dh"))

(constraints
 ((d//dt x dx))
 ((d//dt k dk))
 ((d//dt h dh))

; basic constraints
 (((M + -) x h dx)  (xv hmin 0) (xmsy MSY 0))
 (((M - +) k h dk))
 (((M + + +) x k r h))

; exclude k=0 and h>0
 (((NQ -0 +0) k h) (0 hmin))

; h > r
 (((NQ - +) h r) (hmin hmin))

; exclusion of marginal edges

 ((cornot x h) (xv hmin))
 ((cornot x h) (xv MSY))
 ((cornot x h) (xmin hmin))
 ((cornot x h) (xmin MSY))
 ((cornot x h) (xmsy hmin))
 ((cornot x h) (xmsy MSY))

 ((cornot x r) (xv hmin))
 ((cornot x r) (xmin hmin))
 ((cornot x r) (xmsy hmin))

 ((cornot x dx) (xv 0))
 ((cornot x dx) (xmin 0))
 ((cornot x dx) (xmsy 0))
 ((cornot x dk) (xv 0))
 ((cornot x dk) (xmin 0))
 ((cornot x dk) (xmsy 0))
 ((cornot x dh) (xv 0))
 ((cornot x dh) (xmin 0))
```

```
 ((cornot x dh) (xmsy 0))

 ((cornot h dx) (hmin 0))
 ((cornot h dx) (MSY 0))
 ((cornot h dk) (hmin 0))
 ((cornot h dk) (MSY 0))
 ((cornot h dh) (hmin 0))
 ((cornot h dh) (MSY 0))

 ((cornot r dx) (hmin 0))
 ((cornot r dk) (hmin 0))
 ((cornot r dh) (hmin 0))

 ((cornot dx dk) (0 0))
 ((cornot dx dh) (0 0))
 ((cornot dk dh) (0 0)))

; technical directives

(unreachable-values
  (k kmax) (h hmax) (r 0 rmax))

(weak-cycles)

(with-envisionment)

; initial state of simulation

(make-initial-state
 (x ((xv xmsy) dec))
 (k ((0 kmax) nil))
 (r ((hmin rmax) nil))
 (h ((hmin MSY) nil)))
```

Negotiation Model with Capital and a further restricted qualitative state space. The code contains different constraints for r:

```
(quantity-spaces
 (x (xv xmsy)       "Stock")
 (k (0 kmax)        "Capital")
 (r (0 hmin rmax)   "Catch Recommendation")
 (h (hmin MSY hmax) "Harvest")
 (dx (minf 0 inf)   "dx")
 (dk (minf 0 inf)   "dk")
 (dh (minf 0 inf)   "dh")
 (dr (minf 0 inf)   "dr"))

(constraints
 ((d//dt x dx))
 ((d//dt k dk))
 ((d//dt h dh))
 ((d//dt r dr))

; basic constraints
 (((M + -) x h dx)  (xv hmin 0) (xmsy MSY 0))
 (((M - +) k h dk))
 (((M + + +) x k r h))

; conservative control
; ((M- h dr) (hmin 0))
; qualitative control
  (((M - + -) x dx dh r) (xmsy 0 0 hmin))
  (((M - + -) x k h dh))

; exclude k=0 and h>0
 (((NQ -0 +0) k h) (0 hmin))

; r < h
 (((NQ - +) h r) (hmin hmin))

;; exclusion of marginal edges

 ((cornot x h) (xv hmin))
 ((cornot x h) (xv MSY))
 ((cornot x h) (xmsy hmin))
 ((cornot x h) (xmsy MSY))

 ((cornot x r) (xv hmin))
 ((cornot x r) (xmsy hmin))

 ((cornot x dx) (xv 0))
 ((cornot x dx) (xmsy 0))
 ((cornot x dk) (xv 0))
 ((cornot x dk) (xmsy 0))
 ((cornot x dh) (xv 0))
```

```
 ((cornot x dh) (xmsy 0))
 ((cornot x dr) (xv 0))
 ((cornot x dr) (xmsy 0))

 ((cornot h dx) (hmin 0))
 ((cornot h dx) (MSY 0))
 ((cornot h dk) (hmin 0))
 ((cornot h dk) (MSY 0))
 ((cornot h dh) (hmin 0))
 ((cornot h dh) (MSY 0))
 ((cornot h dr) (hmin 0))
 ((cornot h dr) (MSY 0))

 ((cornot r dx) (hmin 0))
 ((cornot r dk) (hmin 0))
 ((cornot r dh) (hmin 0))

 ((cornot dx dk) (0 0))
 ((cornot dx dh) (0 0))
 ((cornot dk dh) (0 0))
 ((cornot dx dr) (0 0))
 ((cornot dk dr) (0 0))
 ((cornot dh dr) (0 0)))

;;; technical directives

(unreachable-values
  (k kmax) (h hmax) (r 0 rmax))

(weak-cycles)

(with-envisionment)

;;; initial state of simulation

(make-initial-state
 (x ((xv xmsy) dec))
 (k ((0 kmax) nil))
 (r ((hmin rmax) nil))
 (h ((hmin MSY) nil)))
```

Model Code for Lake Management

The basic model with unconstrained phosphorus input L (see section 4.4, p. 116).

```
(quantity-spaces
  (P   (0 l1 l2 eu) "lake phosphorus")
  (M   (0 inf)      "sediment phosphorus")
  (L   (0 inf)      "intake")
  (dP  (minf 0 inf) "dP")
  (dM  (minf 0 inf) "dM")
  (dL  (minf 0 inf) "dL")
  (phi (0 1)        "regime")
  (r   (0 inf)      "recycling"))

(constraints
  ((d//dt P dP))
  ((d//dt M dM))
  ((d//dt L dL))

  (((M + -  ) P M   dM))
  (((M - + +) P r L dP))
  ((mult M phi r) (0 0 0))
  ((S+ P phi) (l1 0) (l2 1))

  ((cornot dP dM) (0 0))
  ((cornot dP dL) (0 0))
  ((cornot dL dM) (0 0)))

(unreachable-values
  (M 0) (L 0) (P 0))

(with-envisionment)

(make-initial-state
  (P ((0 l1) nil))
  (M ((0 inf) nil))
  (L ((0 inf) nil)))
```

Version with qualitative lake management. The quantity space is restricted to $P \geq \lambda_*$.

```
(quantity-spaces
  (P   (l1 l* l2 eu)   "lake phosphorus")
  (M   (0 inf)         "sediment phosphorus")
  (L   (0 inf)         "intake")
  (dP  (minf 0 inf)    "dP")
  (dM  (minf 0 inf)    "dM")
  (dL  (minf 0 inf)    "dL")
  (phi (0 phil* 1)     "regime")
  (r   (0 inf)         "recycling"))

(constraints
  ((d//dt P dP))
  ((d//dt M dM))
  ((d//dt L dL))

  (((M + -  ) P M   dM))
  (((M - + +) P r L dP))
  ((mult M phi r) (0 0 0))
  ((S+ P phi) (l1 0) (l2 1) (l* phil*))
  (((M - + +) P M L dP))

; control
  (((M + - -) P M L dL))

  ((cornot dP dM) (0 0))
  ((cornot dP dL) (0 0))
  ((cornot dL dM) (0 0)))

(unreachable-values
  (M 0) (L 0)  (P l1))

(with-envisionment)

(make-initial-state
  (P ((l* l2) nil))
  (M ((0 inf) nil))
  (L ((0 inf) nil)))
```
