BoF-SvF

NiCu = 'Ni'

#SvF.Resources = [ "abc\_dell" ]

#SvF.useNaN = True

#ObjToReadSols = True

#SvF.solverOptVal ["linear\_solver"] = 'ma86'

SvF.DataPath = '\_DATA100-158/'

CVNumOfIter = -1

InitSols = 0 #True

ShoreErr = False

#RunMode = 'S&S'

RunMode = 'L&P'

SvF.max\_workers = 10

SvF.maxJobs = 3

#SvF.OptStep = [0,0,-0.0001,0, 0.0001, 0,0]

# **The model describes the consequences of Ni mining and processing**

# **by the Pechenganickel plant.**

# **Two atmospheric emission sources** are considered in the towns of:

# **Nickel (Nik) и Zapolyarny (Zap)**.

latNik = 69.413; lonNik = 30.234 # lat - latitude, lon - longitude

latZap = 69.425; lonZap = 30.822

# Time constants:

StartYear = 1946 # Start of plant operation and the start of modeling

EndYear = 2030 # Complete cessation of emissions

StartZap = 1956 # Start of plant operation (emissions) in Zapolyarny

ProgYear = 2050 # End of forecast

Set: t = [ StartYear, EndYear, 1 ] # Modeling interval.

# Model time step - 1 year

rmax = 130 # Radius of modeling area

rStep = 1 # Spatial model step

Set: Fi = [-180, 180, 10]

r = [ 0, rmax, rStep ]

r1 = [ 2\*rStep, rmax, rStep ] # excluding source point

r2 = [ 50, rmax-1, rStep ] #

Set: Prognose = [ StartYear, ProgYear, 1 ]

AzimutInit ( latZap, lonZap ) # Zap in the center

# **Distances and Angles to Sources**

xyNik = LatLonToAzimut ( latNik, lonNik ) # lat,lon -> Cartesian coordinates

def fiNik(X,Y): return degrees ( np.arctan2 (Y-xyNik[1], X-xyNik[0]) ) # fi

def rNik(X,Y): return sqrt ( (Y-xyNik[1])\*\*2 + (X-xyNik[0])\*\*2 ) # r

xyZap = LatLonToAzimut ( latZap, lonZap )

def fiZap(X,Y): return degrees ( np.arctan2 (Y-xyZap[1], X-xyZap[0]) ) # fi

def rZap(X,Y): return sqrt ( (Y-xyZap[1])\*\*2 + (X-xyZap[0])\*\*2 ) # r

# **Atmospheric transport model**:

# deposition is a multiplicative function of three components:

# - dependence on the direction from the source to the point of fallout

# - dependence on the distance from the source to the point of fallout

# - dependence on the intensity of emissions of the source

############################################### Р ROSE

Select Fi, RosePol AS Pfi from NikRosePol37(Fi).sol

# **Wind rose** (the same for both sources) –

# function of direction to the deposition point (угла Fi).

Var: Pfi ( Fi ) > 0.4; Type = Cycle

EQ: ∫( -180, 180, Pfi(Fi)\*d(Fi)) == 360

# source in **Nickel (Nik)**

Var: EmNik ( t ) > 0 # EMISSION in Nickel, tons/year **#2**

# Интенсивность выпадения как функция расстояния до Никеля

Var: PrNik ( r ) > 0; PrNik(0) = PrNik(rStep); PrNik( rmax ) = 0; d/dr(PrNik(r)) <=0

DepNik ( r ) > 0; DepNik( rmax ) = 1 # fraction deposited within radius r

EQ: d/dr1(PrNik(r1)\*r1) <= -0.003 \* PrNik(r1)\*r1

d/dr2(PrNik(r2)\*r2) >= -0.2 \* PrNik(r2)\*r2

DepNik (r) = ∫( 0, r, PrNik(r1)\*r1\*d(r1)) \* 2\*pi

# source in **Zapolyarny**  (Zap)

Var: EmZap ( t ) > 0 # EMISSION in Zapolyarny, tons/year **#2**

Var: PrZap ( r ) > 0; PrZap(0) = PrZap(rStep); PrZap( rmax ) = 0; d/dr(PrZap(r)) <=0

DepZap ( r ) > 0; DepZap( rmax ) = 1

EQ: d/dr1(PrZap(r1)\*r1) <= -0.003 \* PrZap(r1)\*r1

d/dr2(PrZap(r2)\*r2) >= -0.2 \* PrZap(r2)\*r2

DepZap (r) = ∫( 0, r, PrZap(r1)\*r1\*d(r1)) \* 2\*pi

# ∫( 0, rmax, d(r)\*PrZap(r)\*r) \* 2\*pi == 1

############################################### Оба Nik и Zap

EMS2a = Select Year as t, NiNi as EmNikZap from 2Source2030.xlsx # where t > 1976

Var: EmNikZap ( t ) > 0 ; < 700; EmNikZap ( t ) = EmNik ( t ) + EmZap ( t )

Set: t1946\_1955 = [StartYear,**StartZap-1**,1]

tMono = [1955,1976,1]

tMonoNik = [1946,1976,1]

# t1980\_2010 = [1955,2014,1]

t1955\_2014 = [1955,2014,1]

t2010\_2014 = [2010,2014,1]

t2016\_2019 = [2016,2019,1]

t2022\_2030 = [2022,EndYear,1]

t1946\_2009 = [StartYear,2009,1]

EQ:

EmZap( t1946\_1955 ) = 0

d/dtMono(EmZap(tMono)) > 1

EmNik( StartYear ) <= 10

d/dtMonoNik(EmNik(tMonoNik)) > 1

EmNikZap ( t2022\_2030 ) <= 20.

EmZap ( t1955\_2014 ) <= EmNik ( t1955\_2014 ) \* 1.2

d/dt2022\_2030 ( EmNikZap(t2022\_2030)) < 0

# **DEPOSITION** from Nikel and Zapolyarny (mg/m2/year) in x, y, t

# Pollutant can be soluble or insoluble.

Var: SolubleNik( r ) > 0.05; <.5; d/dr (SolubleNik) > 0 # fraction of soluble pollutants.

SolubleZap( r ) > 0.05; <.5; d/dr (SolubleZap) > 0

# Deposition of pollutants (in soluble and insoluble form) from two sources

def Depos\_sol (x,y,t): # Total soluble deposition mg/m2/year **#4b**

dep = 0 #precip \* fonWater /1000 \* 0.4

if t <= EndYear:

if rNik(x,y) <= rmax : dep += Pfi(fiNik (x,y)) \* PrNik(rNik(x,y)) \* \

EmNik(t) \* SolubleNik (rNik(x,y))

if rZap(x,y) <= rmax : dep += Pfi(fiZap (x,y)) \* PrZap(rZap(x,y)) \* \

EmZap(t) \* SolubleZap (rZap(x,y))

return dep \* 1000

def Depos\_insol (x,y,t): # Total insoluble deposition mg/m2/year **#4a**

if t > EndYear: return 0

dep = 0

if rNik(x,y) <= rmax : dep += Pfi(fiNik (x,y)) \* PrNik(rNik(x,y)) \* EmNik(t) \* \

(1-SolubleNik (rNik(x,y)))

if rZap(x,y) <= rmax : dep += Pfi(fiZap (x,y)) \* PrZap(rZap(x,y)) \* EmZap(t) \* \

(1-SolubleZap (rZap(x,y)))

return dep \* 1000

CONS = Select nP, X,Y, Catch, S as SQw, Catch as SQs, lake1, peat1 from POINT.xlsx

CONS.SQw[:] += (CONS.Catch[:]-CONS.SQw[:]) \* (CONS.lake1[:]+CONS.peat1[:])/100

CONS.SQs[:] -= CONS.SQw[:]

# A total of **108 points** are considered in the modeling region.

Set: nP = [ , , 1] # nP = [ 1, 108, 1]

# These are the points where measurements are available/

# It is assumed that at each point (number nP) there is a catchment area.

# Watershed parameters:

Param: X( nP ) # X coordinate (km)

Y( nP ) # Y coordinate (km)

Catch( nP ) # Catchment area (km2)

SQw( nP ) # Lake area (km2)

SQs( nP ) # Land area (km2) SQs(nP) = Catch(nP) - SQw(nP)

# Common parameters for all catchments:

precip = 0.4 # precipitation m/year

fonWater = 1.0 # background pollution of water mkg/dm3 = mg/m3

fonSoilDW = 10 # background pollution of soil mg/kg(сух.веса)

fonSedi = 20 # background pollution of bottom sediments mg/kg(сух.веса)

depISout = 0.2 # Fraction of pollutants draining from the catchment

intoIS = 0.2 # proportion of solubles bound (into insolubles) in the soil

Var: Snow <0.5; >0.1 # Fraction of deposition entering the lake with snowmelt

notSnow; notSnow = 1-Snow # Fraction of deposition remaining on land

# For each point nP we define a **WATERSHED MODEL** with a lake –

# 3-media model: land, water and bottom sediments.

# Soluble and insoluble depositions (mg/m2/year) on the watershed nP

Var: depIS ( nP, t ) #>=0; # insoluble depositions mg/m2/year

EQ: depIS = Depos\_insol (X(nP),Y(nP),t)

Var: depS ( nP, t ) #> 0; # soluble depositions mg/m2/year

EQ: depS = Depos\_sol (X,Y,t)

# **SOIL**. Dynamics of soil reserves

Var: hSoil > 0.005; #< 0.04 #35 # Layer thickness in meters

Var: S ( nP , t ) > 7.5 # amount of pollutant in soil mg/m2 **#5**

leaching > 0.0015; # leaching coefficient:

# fraction of insolubles converted to soluble

EQ: S(nP,StartYear) = fonSoilDW \* hSoil\*800 # initial condition

# (800\*hSoil) - conversion from content (in dry weight) to reserves

d/dt ( S ) = (depIS(nP,t)+depS(nP,t)\*intoIS) \* notSnow - leaching \* S

# change in stock = precipitation on (unfrozen) soil - leaching

# It is assumed that at the initial moment the reserves in the soil correspond to background values, that only insoluble forms that fell during the snowless period or bound soluble forms remain in the soil, that the leaching process converts Ni into a soluble form, which is removed from the soil into water.

# **BOTTOM SEDIMENTS** Stock dynamics

Var: B ( nP , t ) > 0 # amount in bottom sediments mg/m2 **#6**

botSedLeach > 0.00; botSedLeach = leaching

# leaching coefficient of bottom sediments

EQ: B(nP,StartYear) = 0 # initial condition

d/dt ( B ) = depIS(nP,t) \*(1+SQs(nP)/SQw(nP)\*Snow) \*(1-depISout) - \

botSedLeach\*B + fonSedi\*0.1 # Vsedim\*0.2=0.1

# change in stock = input minus runoff - leaching + background

# It is assumed that bottom sediments are formed from insoluble forms falling directly on the lake surface, from insoluble forms falling directly on land in winter (and transferred to the lake during floods) and from a background value reflecting pre-industrial processes of bedrock weathering and soil leaching.

# **WATER**. Dynamics of the content (concentration) of pollution in lake water

# quantity = water\*S\*H + depS\*Catch + leaching·S·SQs(nP) ...

# volume = S\*H + Catch\*proc

# concentration = quantity/volume

Var: W ( nP , t ) > 0 # pollutant concentration mkg/dm3 = mg/m3 **#7**

H > 0; < 2 #20\*precip # depth (m)

EQ: W(nP,StartYear) =fonWater # initial condition

d/dt( W ) = ( depS(nP,t)\*(SQw(nP)+SQs(nP)\*(Snow+notSnow\*(1-intoIS))) \

+ leaching·S·SQs(nP) + botSedLeach·B·SQw(nP) \

– Catch(nP)·precip·(W-fonWater) ) / ( SQw(nP)\*H+Catch(nP)\*precip )

# change in concentration =

# input with precipitation, from soil and bottom sediments - removal with runoff

# It is assumed that soluble forms (except for those retained in the soil) enter the lake together with all sediments (precip = 0.40 m/year), which all soluble forms formed in the soil as a result of leaching are added to the lake water and some is carried away with the runoff.

############################################################## # Проекторы **СВЯЗЬ С ИЗМЕРЕНИЯМИ**

UseNaN = True

TDep = Select nP AS nPD, Year as tD, ROWNUM as nD, depNi as deposD, X as Xd,Y as Yd from EXP\_DEPOS.xlsx # ВЫПАДЕНИЯ

###### DEPOSITION mg/m2/год 1e-3 or t/km2/year or kg /km2 /year #####

Set: nD = [,,1]

Param: nPD (nD)

tD (nD)

Var: deposD (nD) > 0 # DEPOSITION mg/m2/год

EQ: deposD (nD) = depIS(nPD(nD),tD(nD)) + depS(nPD(nD),tD(nD))

#---------------------# mg/kg(сух.веса): soilD\*0.8(кг.сух/литр)\*10\*10\*10 (mg/m3) \* hSoil(m) = soilD\*800\*hSoil(m) = S (mg/m2) ПОЧВА

TSoil = Select nP AS nPS, Year as tS, ROWNUM as nS, soilNi as **soilD**, X as Xs,Y as Ys from EXP\_SOIL.xlsx

**for i in TSoil.sR : TSoil.soilD[i] = max (TSoil.soilD[i],fonSoilDW)**

Set: nS = [,,1]

Param: nPS (nS)

tS (nS)

Var: soilD (nS) > 0 # СОДЕРЖАНИЕ mg/kg(сух.веса)

# FonsoilD (nS) > 0 # Фоновое значение, начальное условие

EQ: soilD \* 800\*hSoil = S ( nPS , tS )

#--------------------------------------------------------------------------------------------------------------- # WATER

TWater = Select X as Xw,Y as Yw, Year AS tW, Avg-TNi AS waterD, ROWNUM as nW, nP AS nPW, nP, CONn from EXP\_LAKE.xlsx

#MakeSets\_byParam nPW 7

CV: CV\_NumSets=7; CV\_Unit ='nPW'

Set: nW = [,,1]

Param: nPW (nW)

tW (nW)

Var: waterD (nW) > 0 # mkg/dm3 ВОДА

# wFon > 0

#EQ: waterD = Depos\_sol ( X(nPW(nW)), Y(nPW(nW)), tW(nW) ) / precip + wFon

EQ: waterD = W ( nPW , tW )

#\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_# ДО SEDIMENTION

# B(mg/kg-dw) \*0.25(kg-dw/dm3) \*10+3(dm3/m3) \*Vsedim(mm/year) 10-3(m/mm) = botSedD\*Vsedim\*0.25(mg/m2/year) = B (mg/m2/year) - поступление

TBott = Select nP AS nPB, Year as tB, ROWNUM as nB, Vsedim, delT, Ni\_0\_1 as botSedD, X as Xb,Y as Yb from EXP\_SEDIMNT.xlsx

#for ii in TBott.sR :

# TBott.delT[ii] = min (TBott.delT[ii],2)

Set: nB = [,,1]

Param: nPB (nB)

tB (nB)

Vsedim(nB) # mm/year

delT (nB)

Var: botSedD (nB) > 0 # СОДЕРЖАНИЕ mg/kg(сух.веса)

sedimMult > 0

EQ: botSedD(nB)\*Vsedim(nB)\* **sedimMult\*0.25** = ( B(nPB,tB(nB)) - B(nPB,tB(nB)-delT(nB)) ) / delT(nB)

#########################################################################################################

def TimeStep(Xi,Yi,ti,Catchi,Si, s,w,b):

SQsi = Catchi-Si

depSo = Depos\_sol ( Xi,Yi,ti )

depISo = Depos\_insol ( Xi,Yi,ti )

Bo = b + depISo \* (1 + SQsi/Si\*Snow )\*(1-depISout) **-botSedLeach**\*b +fonSedi \* 0.1

So = s \* (1 - leaching ) + (depISo+depSo\*intoIS) \* notSnow

# Wo = w + ( depSo\*(Si+SQsi\*(Snow+notSnow\*(1-intoIS)))+leaching·s·SQsi + botSedLeach·b·Si -Catchi\*precip\*w )/(Si\*H+Catchi\*precip)

Wo = w + ( depSo\*(Si+SQsi\*(Snow+notSnow\*(1-intoIS)))+leaching·s·SQsi + botSedLeach·b·Si -Catchi\*precip\*(w-fonWater) )/(Si\*H+Catchi\*precip)

return depSo, depISo, Bo, So, Wo

def RunModel ( start, end ) :

if start == StartYear :

S.grd[:,0] = fonSoilDW \* hSoil\*800 #fonSoil

W.grd[:,0] = fonWater

B.grd[:,0] = 0

for bi in nP.NodS:

bv = nP.Val[bi]

for tv in range (start, end+1, 1) :

ti = tv - StartYear

# print (ti)

**depS.grd[bi,ti],depIS.grd[bi,ti],Bo, So, Wo = TimeStep(X(bv),Y(bv),tv, Catch(bv),SQw(bv), \**

**S(bv,tv),W(bv,tv),B(bv,tv))**

if tv < end :

S.grd[bi,ti+1] = So

W.grd[bi,ti+1] = Wo

**B.grd[bi,ti+1] = Bo**

if (InitSols):

Task.ReadSols ()

# DrLake (71)

RunModel ( StartYear, EndYear )

#Draw depIS

#Draw depS

#Draw B

#Draw soil

#Draw water

for n in nD.NodS:

nv = nD.Val[n]

deposD.grd[n] = depIS(nPD(nv),tD(nv)) + depS(nPD(nv),tD(nv))

for n in nS.NodS:

nv = nS.Val[n]

soilD.grd[n] = S ( nPS(nv) , tS(nv) ) / (800\*hSoil)

for n in nB.NodS:

nv = nB.Val[n]

# botNi.grd[n] = ∫( tB(nv)-delT(nv), tB(nv), d(T)\*bNi(bB(nv),T) ) / delT(nv) / (Vsedim(nv)\*0.25)

# botSedD.grd[n] = ∫( tB(nv)-delT(nv), tB(nv), d(t)\*botSed(nPB(nv),t) ) / delT(nv) / (Vsedim(nv)\*sedimMult\*0.25)

botSedD.grd[n] = ( B(nPB(nv),tB(nv)) - B(nPB(nv),tB(nv)-delT(nv)) ) / delT(nv) / (Vsedim(nv)\***sedimMult\*0.25)**

# ( botSed(nPB,tB(nB)) - botSed(nPB,tB(nB)-delT(nB)) ) / delT(nB)

# botSedD(nB)\*Vsedim(nB)\* sedimMult\*0.25 = ∫( tB(nB)-delT(nB), tB(nB), d(t)\*botSed(nPB,t) ) / delT(nB) /()

# Draw botSedD

Task.SaveSols ()

**Obj:** waterD.MSDrel(2.0) + 0.0002\*deposD.MSD\_no\_mu() + Penal[4]\*EmNikZap.MSD\_no\_mu() + Penal[5]\*soilD.MSDrel\_no\_mu(30.0) \

+ Penal[7] \* Pfi.MSD\_no\_mu() +Penal[6]\*botSedD.MSDrel\_no\_mu(40) \

+ Pfi.ComplCyc0E (Penal[0]) + DepNik.Compl (Penal[1]) + EmNik.ComplMean2 (Penal[2]) \

+ DepZap.Compl (Penal[1]) + EmZap.ComplMean2 (Penal[2]) \

+ SolubleNik.ComplMean2 (Penal[3]) + SolubleZap.ComplMean2 (Penal[3])

INCLUDE: Draw/Draw-09.mng

**EoF**