Moduł 12
Przypadek użycia 1
COVID Data hub

iBigWorld:
Innovations for Big Data in a Real World

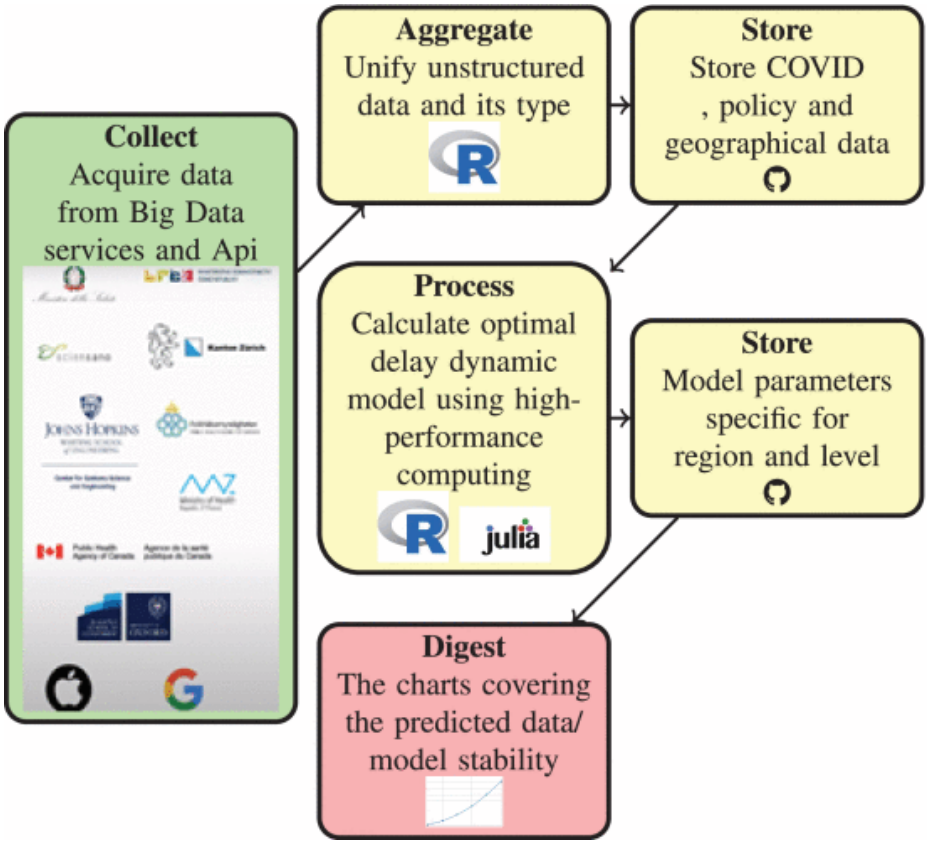
Zespół UBB

iBigWorld:2020-1-PL01-KA203-082197

Co-funded by the Erasmus+ Programme of the European Union

Potok Big Data

- COVID-19 Big Data Hub umożliwia nam przewidywanie rozwoju pandemii z uwzględnieniem wielu szczepów wirusa i opóźnień zakaźności
- Modele dynamiczne z dwoma szczepami z rozproszonymi opóźnieniami zostały dopasowane do szeregów czasowych pobranych z centrum danych COVID
- Wykorzystano dane na poziomie krajowym, regionalnym i powiatowym, które są płynnie zintegrowane z otwartymi danymi Banku Światowego, raportami mobilności Google, raportami mobilności Apple
- Identyfikacja parametrów została zrealizowana za pomocą algorytmu COBYLA
- Symulacje zostały zaimplementowane za pomocą modułu Julia.



iBigWorld:2020-1-PL01-KA203-082197

Co-funded by the Erasmus+ Programme of the European Union

Pobranie danych

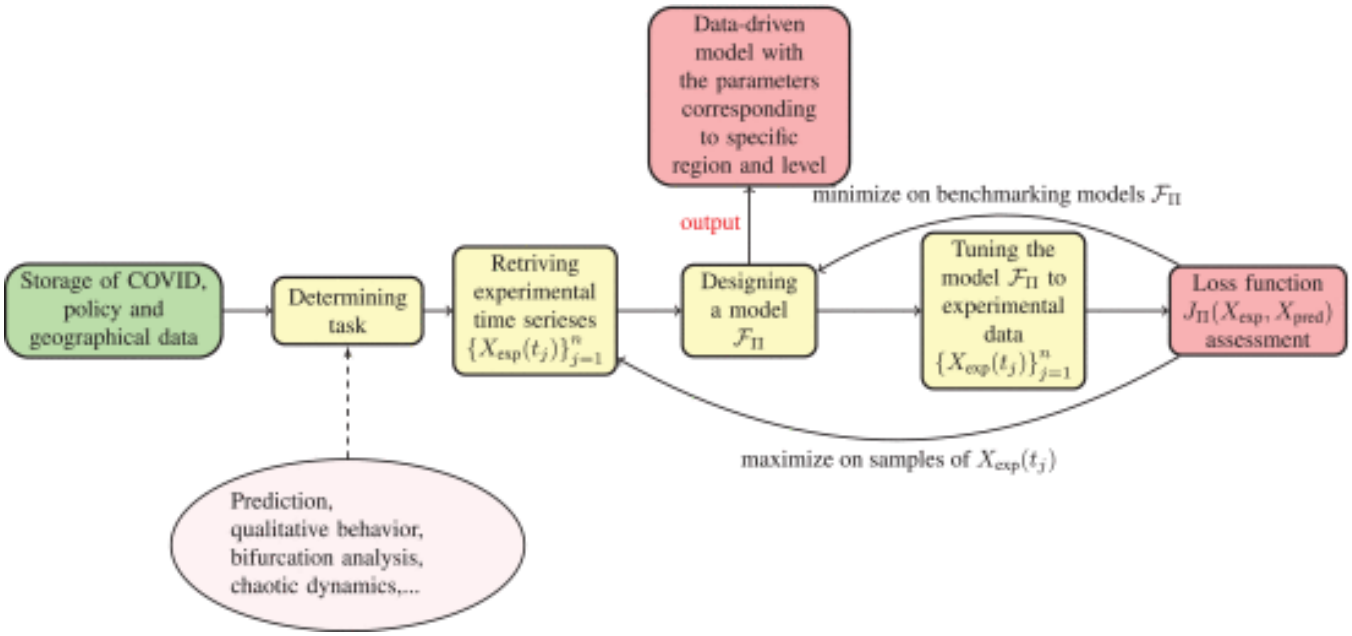
- library("COVID19")

- ###COUNTRY###############
- x <- covid19(country = countryname, level = 1)
- ###REGION################
- x <- covid19(country = countryname, level = levelname)
- x <- x[which(x$administrative_area_level_2=="Texas"),]
- ###COUNTY#################
- x <- covid19(country = countryname, level = levelname)
- x <- x[which(x$administrative_area_level_2=="Rheinland-Pfalz" & x$administrative_area_level_3=="SK Speyer"),]

Co-funded by the Erasmus+ Programme of the European Union

Potok Big Data

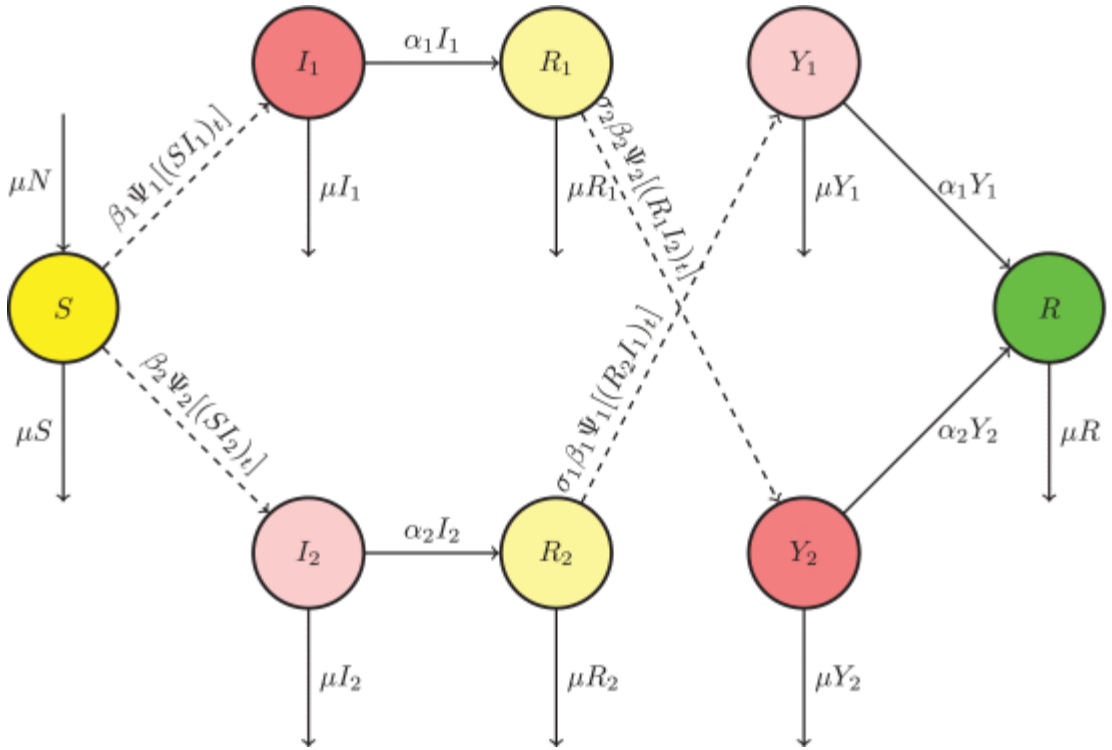
- Potok rozwiązania problemu SciML



Co-funded by the Erasmus+ Programme of the European Union

Model

- Strojenie modelu



iBigWorld:2020-1-PL01-KA203-082197

Co-funded by the Erasmus+ Programme of the European Union

Potok Big Data

- Strojenie modelu

Input data: X_{exp}, Π_{lower}, Π_{upper}, Π_{init}
Result: Π_{opt}
form the initial simplex C_{init} with the vertices
$\Pi_0^{\text{init}}, \Pi_1^{\text{init}}, \ldots, \Pi_{27}^{\text{init}}$;
repeat
  for the current simplex C calculate the values
  $\hat{\Phi}_C(\Pi_i), i = \overline{0, 27}$;
  search the vertex Π_l determined by the equation
  $\hat{\Phi}_C(\Pi_l) = \min\left\{\hat{\Phi}_C(\Pi_i), i = \overline{0, 27}\right\}$;
  calculate new vertex as
  $\Pi_{new} := -\theta\Pi_l + (1+\theta)\frac{1}{27}\sum_{i=\overline{0,27}, i\neq l}\Pi_i$, where
  reflection coefficient $\theta \in (0, 1)$ being chosen as
  small as possible in order $\hat{\Phi}_C(\Pi_l)$ not were the
  least calculated function value so far;
  form modified simplex C_{new} replacing vertex Π_l
  with Π_{new};
  search Π_{opt} as a solution of the problem of linear
  optimization

$$\left.\begin{array}{l}\texttt{minimize} \quad \hat{\Phi}_{C_{\text{new}}}(\Pi), \\ \texttt{subject to}\ \Pi \in C_{\text{new}}\end{array}\right\} \qquad (16)$$

until stop condition;
return Π_{opt};

iBigWorld:2020-1-PL01-KA203-082197

Co-funded by the
Erasmus+ Programme
of the European Union

Potok Big Data

- Strojenie modelu

Input data: X_{exp}, Π_{lower}, Π_{upper}, Π_{init}
Result: Π_{opt}
form the initial simplex C_{init} with the vertices
$\Pi_0^{\text{init}}, \Pi_1^{\text{init}}, \ldots, \Pi_{27}^{\text{init}}$;
repeat
  for the current simplex C calculate the values
  $\hat{\Phi}_C(\Pi_i), i = \overline{0, 27}$;
  search the vertex Π_l determined by the equation
  $\hat{\Phi}_C(\Pi_l) = \min\left\{\hat{\Phi}_C(\Pi_i), i = \overline{0, 27}\right\}$;
  calculate new vertex as
  $\Pi_{new} := -\theta\Pi_l + (1+\theta)\frac{1}{27}\sum_{i=\overline{0,27}, i\neq l}\Pi_i$, where
  reflection coefficient $\theta \in (0, 1)$ being chosen as
  small as possible in order $\hat{\Phi}_C(\Pi_l)$ not were the
  least calculated function value so far;
  form modified simplex C_{new} replacing vertex Π_l
  with Π_{new};
  search Π_{opt} as a solution of the problem of linear
  optimization

$$\left.\begin{array}{l}\texttt{minimize} \quad \hat{\Phi}_{C_{\text{new}}}(\Pi), \\ \texttt{subject to}\ \Pi \in C_{\text{new}}\end{array}\right\} \qquad (16)$$

until stop condition;
return Π_{opt};

iBigWorld:2020-1-PL01-KA203-082197

Co-funded by the
Erasmus+ Programme
of the European Union

Wysoko wydajne operacje w Julia

```
f <- JuliaCall::julia_eval("function f(du, u, h, p, t)
  mu=p[1]
  beta_1=p[2]
  alpha_1=p[3]
  a_1=p[4]
  m_1=p[5]
  tau_m_1=p[6]
  beta_2=p[7]
  alpha_2=p[8]
  a_2=p[9]
  m_2=p[10]
  tau_m_2=p[11]
  sigma_1=p[12]
  sigma_2=p[13]
  n=100
  c=0.01
  rmse_1=sqrt((m_1+1)/a_1^2)
  tau_M_1_tilde=sqrt(rmse_1^2 / c)
  tau_M_1=tau_m_1+(m_1+1)/a_1+tau_M_1_tilde
  rmse_2=sqrt((m_2+1)/a_2^2)
  tau_M_2_tilde=sqrt(rmse_2^2 / c)
  tau_M_2=tau_m_2+(m_2+1)/a_2+tau_M_2_tilde
  lags_1=range(1/n, length=n, stop=tau_M_1*(1-1/n))
  lags_2=range(1/n, length=n, stop=tau_M_2*(1-1/n))
```


Co-funded by the
Erasmus+ Programme
of the European Union

Wysoko wydajne operacje w Julia (kontynuacja)

```
N=p[14]
S_0=p[15]
I_1_0=p[16]
R_1_0=p[17]
I_2_0=p[18]
R_2_0=p[19]
Y_1_0=p[20]
Y_2_0=p[21]
beta_1_tilde=p[22]
omega_beta_1=p[23]
beta_2_tilde=p[24]
omega_beta_2=p[25]
alpha_1_tilde=p[26]
omega_alpha_1=p[27]
alpha_2_tilde=p[28]
omega_alpha_2=p[29]
du[1]= mu*(N-u[1]) - beta(beta_1,beta_1_tilde,omega_beta_1,t)*u[1]*u[2] - beta(beta_2,beta_2_tilde,omega_beta_2,t)*u[1]*u[4]
du[2]=beta(beta_1,beta_1_tilde,omega_beta_1,t)*tau_M_1*(1/n)*sum([psi(a_1,m_1,tau_m_1,tau)*h(p,t-tau)[1]*h(p,t-tau)[2] for tau in lags_1]) - mu*u[2] - alpha(alpha_1,alpha_1_tilde,omega_alpha_1,t)*u[2]
du[3]=alpha(alpha_1,alpha_1_tilde,omega_alpha_1,t)*u[2] - (mu+sigma_2*beta(beta_2,beta_2_tilde,omega_beta_2,t)*u[4])*u[3]
du[4]=beta(beta_2,beta_2_tilde,omega_beta_2,t)*tau_M_2*(1/n)*sum([psi(a_2,m_2,tau_m_2,tau)*h(p,t-tau)[1]*h(p,t-tau)[4] for tau in lags_2]) - mu*u[4] - alpha(alpha_2,alpha_2_tilde,omega_alpha_2,t)*u[4]
du[5]=alpha(alpha_2,alpha_2_tilde,omega_alpha_2,t)*u[4] - (mu+sigma_1*beta(beta_1,beta_1_tilde,omega_beta_1,t)*u[2])*u[5]
du[6]=sigma_1*beta(beta_1,beta_1_tilde,omega_beta_1,t)*tau_M_1*(1/n)*sum([psi(a_1,m_1,tau_m_1,tau)*h(p,t-tau)[5]*h(p,t-tau)[2] for tau in lags_1]) - mu*u[6] - alpha(alpha_1,alpha_1_tilde,omega_alpha_1,t)*u[6]
du[7]=sigma_2*beta(beta_2,beta_2_tilde,omega_beta_2,t)*tau_M_2*(1/n)*sum([psi(a_2,m_2,tau_m_2,tau)*h(p,t-tau)[3]*h(p,t-tau)[4] for tau in lags_2]) - mu*u[7] - alpha(alpha_2,alpha_2_tilde,omega_alpha_2,t)*u[7]
end")
```

iBigWorld:2020-1-PL01-KA203-082197

Co-funded by the Erasmus+ Programme of the European Union

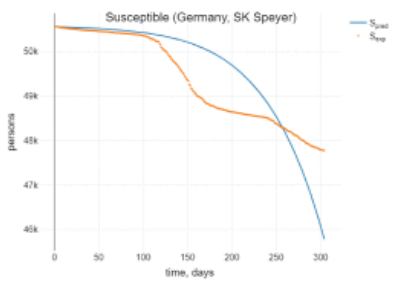
Wysoko wydajne operacje w Julia (kontynuacja)

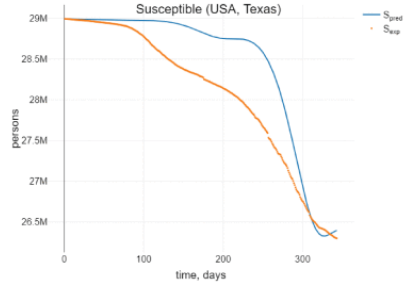
```
h <- JuliaCall::julia_eval("function h(p, t)
  [p[15],p[16],p[17],p[18],p[19],p[20],p[21]]
end")
psi <- JuliaCall::julia_eval("function psi(a, m, tau_m, tau)
  if tau <= tau_m
    res = 0
  else
    res = (a^(m+1)/gamma(m+1))*(tau-tau_m)^m * exp(-a*(tau-tau_m))
  end
  res
end")
beta <- JuliaCall::julia_eval("function beta(beta_i,beta_i_tilde,omega_beta_i,t)
  beta_i + beta_i_tilde*sin(2*pi*t/omega_beta_i)
end")
alpha <- JuliaCall::julia_eval("function alpha(alpha_i,alpha_i_tilde,omega_alpha_i,t)
  alpha_i + alpha_i_tilde*sin(2*pi*t/omega_alpha_i)
end")
```

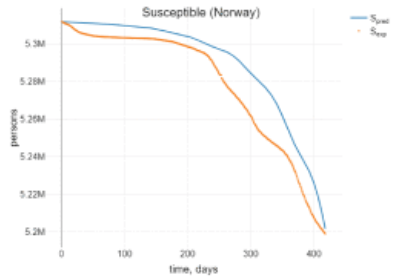
iBigWorld:2020-1-PL01-KA203-082197

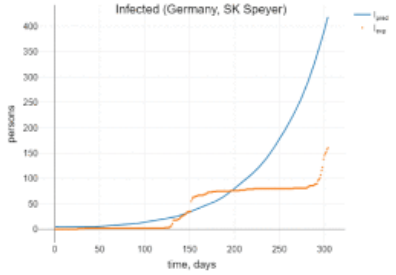
Co-funded by the Erasmus+ Programme of the European Union

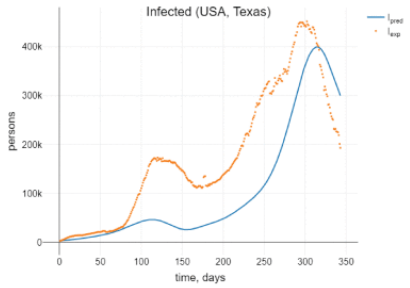
Wizualizacja

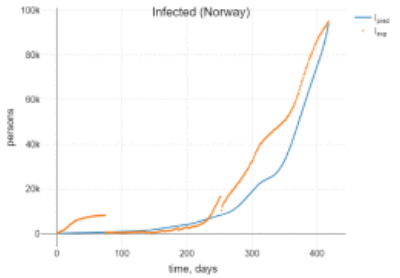


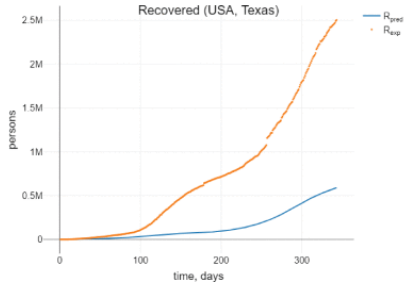


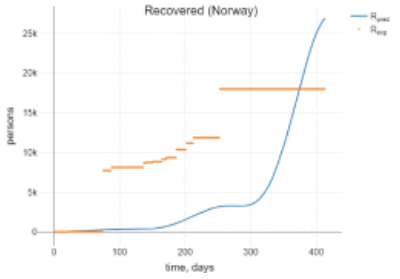












iBigWorld:2020-1-PL01-KA203-082197

Co-funded by the
Erasmus+ Programme
of the European Union