Optymalna strategia podawania leku

Tomasz Kanas

13 czerwca 2020

Zauważmy, że rozwiązanie V_1, V_2, K zależą od wyboru funkcji g, którą nazywamy sterowaniem. W tym modelu g interpretujemy jako...

1 Sformułowanie problemu

Celem pracy jest znalezienie strategii podawania leku, przy leczeniu nowotworu, pozwalającej osiągnąć możliwie największą skuteczność terapii. W tym celu skorzystamy z modelu przedstawionego w pracy [tu wypada zacytować wyjściową pracę]. Model ten przedstawia rozwój nowotworu w czasie w zależności od dawki leku za pomocą równania różniczkowego:

$$\begin{aligned} V_1'(t) &= \lambda_1 V_1 F\left(\frac{V_1+\alpha_{12}V_2}{K}\right) - \beta_1 V_1 g(t), \\ V_2'(t) &= \lambda_2 V_2 F\left(\frac{V_2+\alpha_{21}V_1}{K}\right) - \beta_2 V_2 g(t), \\ K'(t) &= -\mu K + (V_1+V_2) - d(V_1+V_2)^{2/3}K - \beta K g(t) \end{aligned} \tag{1}$$

dla $t \in [0,T]$, z warunkami początkowymi

$$V_1(0) = V_{10},\ V_2(0) = V_{20},\ K(0) = K_0 \tag{2}$$

gdzie $F(x) = -\ln(x)$, $0 \le g(t) \le g_{\max}$, oraz

$$\lambda_1, \lambda_2, \alpha_{12}, \alpha_{21}, \beta_1, \beta_2, \beta, \mu, d, V_{10}, V_{20}, K_0 \le 0$$

naprawdę wszystkie ujemne ?

są zadanymi parametrami.

Funkcja $V_1(t)$ modeluje liczbę komórek guza podatnych na lek w momencie t, $V_2(t)$ liczbę komórek guza odpornych na lek, a $K(t)$ jest parametrem nazwanym w pracy "uczynnieniem".

Zadanie polega na znalezieniu mierzalnej funkcji $g : [0,T] \to [0, g_{\max}]$ takiej, że rozwiązanie (1) minimalizuje funkcjonał

tylko ?!

$$J(g) = \cancel{J(V_1,V_2,K) =} \int_0^T V_1(t) + V_2(t)dt + \omega \int_0^T G\left(\frac{V_2(t)-V_1(t)}{\epsilon}\right)dt \tag{3}$$

gdzie V_1, V_2 są rozwiązaniami (1) dla g oraz

$$G(x) = \frac{1+\tanh(x)}{2} \quad \omega, \epsilon > 0$$

Proponuję zdef. "Problem 1"

Problem ten w literaturze nazywa się problemem optymalnego sterowania, ~~a funkcję g sterowaniem.~~

Zauważmy, że nie wymagamy of g nawet ciągłości, więc rozwiązanie (1) może nie istnieć. Oczywiście w zadaniu interesują nas tylko takie sterowania g dla których rozwiązanie (1) istnieje i jest jednoznaczne. ~~Można wtedy przydzielić sterowaniu wartość funkcjonału $J(g) = J(V_1, V_2, K)$, gdzie V_1, V_2, K są rozwiązaniem (1).~~ Zauważmy też, że funkcja F posiada osobliwość w 0, ale nie jest to dla nas problemem, ponieważ $\alpha_{12}, \alpha_{21} > 0$, więc osiągnięcie tej osobliwości nastąpi tylko gdy $V_1 = V_2 = 0$, czyli gdy pacjent nie ma żadnych komórek nowotworowych, więc jest wyleczony i możemy zakończyć terapię.

to jak można wtedy liczyć (3)?

Podsumowując, celem pracy ? jest znalezienie funkcji mierzalnej $g : [0,T] \to [0, g_{\max}]$~~, oraz funkcji $V_1, V_2, K : [0,T] \to [0,\infty)$ spełniających (1) i (2), oraz~~ minimalizującej funkcjonał celu (3).

to nie jest dla mnie jasne. Myślę, że dla $K \approx 0$ i $V_1, V_2 \approx 0$ i tak możemy mieć problemy. Proponuję to zauważyć, ale nie zajmować (dopóki główny temat nie jest skończony)

Rozwiązując zadanie (*) przybliżamy rozwiązanie pełnego zadania ...

1.1 Problem przybliżony

Analityczne rozwiązywanie problemu optymalnego sterowania rzadko kiedy jest możliwe, a nawet gdy jest możliwe, jest trudne. Z tego powodu zdecydujemy się na szukanie rozwiązania przybliżonego. Celem pracy jest ---

~~Najpierw ograniczmy problem do problemu~~ optymalizacji skończenie wymiarowej. W tym celu ~~wprowadźmy dyskretyzacje czasu, poprzez~~ ustalenie my siatki g dyskretyzacji, ~~czyli ciągu punktów w czasie~~: przedziału [0,T]:

$$0 = t_0 < t_1 < \dots < t_{n-1} < t_n = T \tag{4}$$

Możemy teraz przybliżać sterowanie za pomocą splajnu ~~z węzłami~~ interpolującego g w punktach t_i. ~~Z powodu istnienia ograniczeń na wartości sterowania $0 \le g(t) \le g_{\max}$~~, Dla prostoty ograniczymy się do splajnów stopnia 0 i 1, ~~ponieważ zapewnienie spełnienia tych ograniczeń dla splajnów dowolnego stopnia jest trudniejsze i wymaga więcej obliczeń~~. Ostatecznie przybliżone sterowanie ma postać:

$$\hat{g}(t) = g_i \text{ gdy } t \in [t_{i-1}, t_i) \tag{5}$$

lub

$$\hat{g}(t) = \frac{(t_i - t)g_i + (t - t_{i-1})g_{i-1}}{t_i - t_{i-1}} \text{ gdy } t \in [t_{i-1}, t_i) \tag{6}$$

i jest jednoznacznie zdefiniowane przez wartości $g_i = \hat{g}(t_i)$, $i = 0..n$.

Zauważmy, że przy sterowaniu (5), prawa strona (1) nie jest ciągła, ale jest różniczkowalna i Lipszycowska na każdym przedziale, więc z tw. Picarda-Lindelöfa o istnieniu i jednoznaczności rozwiązań, rozwiązanie (1) istnieje na każdym przedziale $[t_{i-1}, t_i]$ i jest jednoznacznie wyznaczone przez watość w punkcie początkowym. Dla pierwszego przedziału punkt początkowy mamy zadany przez (2), dla i-tego przedziału punkt początkowy jest jednoznacznie zadany przez rozwiązanie dla poprzedniego przedziału, więc rozwiązanie na całym $[0, T]$ jest określone jednoznacznie. ?!

Mając tak przybliżoną funkcję sterowania możemy numerycznie przybliżyć rozwiązanie równania różniczkowego (1). Użyjemy do tego metody Rungego-Kutty rzędu r ze stałym krokiem długości h. Dla uproszczenia notacji oznaczmy (1) przez } red.

warunek początkowy

Jak Pan zmieni szyk zdania, to nie będzie nieporozumień.

$$\dot{y}(t) = f(t, y, g), \;\; y = \begin{pmatrix} V_1 \\ V_2 \\ K \end{pmatrix}, \;\; f = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix} \tag{7}$$

wtedy przybliżone rozwiązanie (7) w punkcie $t_n + (a+1)h$ wyraża się przez:

$$\begin{aligned} k_1 &= f(t_n + ah, \hat{y}(t_n + ah), \hat{g}) \\ k_l &= f(t_n + c_l h, \hat{y}(t_n + ah) + h\sum_{i=1}^{l-1} a_{li} k_i, \hat{g}) \\ \hat{y}(t_n + (a+1)h) &= \hat{y}(t_n + ah) + h\sum_{i=1}^{r} b_i k_i \end{aligned} \tag{8}$$

co to? proponuję przemyśleć notację.

czym jest "a"?

gdzie c_l, a_{li}, b_i są stałymi zależnymi od wybranej metody.

Zostało już tylko przybliżyć funkcjonał celu (3). Zapiszmy go w postaci

$$J(y) = \int_0^T j(y(t))dt \tag{9}$$

to nie wynika z tw. P-L! powinniśmy o tym porozmawiać.

gdzie

$$j(y(\cancel{t})) = V_1 + V_2 + \omega G\left(\frac{V_2 - V_1}{\epsilon}\right),\ y(\cancel{t}) = \begin{pmatrix} V_1 \\ V_2 \\ K \end{pmatrix} \tag{10}$$

wtedy ogólny wzór na kwadraturę ze stałym krokiem h_1 przybliżającą (9) to

$$\hat{J}(\hat{y}) = h_1 \sum_{i=0}^{N} \alpha_i j(\hat{y}(ih_1)) \tag{11}$$

może warto wziąć $h_1 = h$ (dlaczego?)

gdzie $N = \frac{T}{h_1}$, a α_i są stałymi zależnymi od wybranej kwadratury.

W ten sposób wyraziliśmy przybliżoną funkcję celu jako funkcję $g_1, \ldots, g_n$, więc problem przybliżony sprowadza się do

proponuję zdef. "Problem 2".

(*)

$$\min_{g_1,\ldots,g_n} \hat{J}(g_1, \ldots, g_n) \text{ z ograniczeniami} \tag{12}$$

$$\forall_{i\in\{1,\ldots,n\}} 0 \le g_i \le g_{\max} \tag{13}$$

red. np. $\hat{J}$ w (11) jest na $\mathbb{R}^3$.

Jest to problem optymalizacji nieliniowej z ograniczeniami i istnieją implementacje metod pozwalających uzyskać przybliżone rozwiązanie tego problemu. To podejście do numerycznego problemu optymalnego sterowania jest podobne do zaproponowanego w [1] i nazywa się sekwencyjnym (ang. "sequential").

dlaczego? co w (*) jest "sekwencyjnego"?

1.2 Plan rozwiązania

Aby obliczyć wynik problemu przybliżonego skorzystamy ze środowiska MATLAB/Octave wraz z dostarczonym z nim optymalizatorem problemu optymalizacji nieliniowej z ograniczeniami (FMINICON). W tym celu zaimplementujemy przejście od problemu optymalnego sterowania. Aby poprawić złożoność czasową optymalizacji i tym samym umożliwić stosowanie gęstszej siatki dyskretyzacji, obliczymy gradient przybliżonej funkcji celu. Będziemy też musieli znaleźć odpowiednią siatkę dyskretyzacji i punkt startowy dla optymalizatora.

2 Implementacja

Optymalizator FMINICON wymaga dostarczenia funkcji do optymalizowania, czyli w naszym przypadku funkcji $\hat{J}(g_1, \ldots, g_n)$, oraz ograniczeń, czyli w naszym przypadku jedynie (13). Optymalizator umożliwia też zwracanie gradientu przez optymalizowaną funkcję. Domyślnie optymalizator wyznacza ten gradient za pomocą różnic skończonych, co wymaga uruchomienia funkcji celu liniowo wiele razy względem liczby parametrów. Aby poprawić wydajność optymalizacji zaimplementujemy liczenie gradientu przybliżonej funkcji celu ze względu na parametry $g_1, \ldots, g_n$.

?

Zauważmy, że znalezienie dobrego wyniku będzie wymagało zapewne wielokrotnego wywołania naszej funkcji celu przez optymalizator, więc zależy nam aby funkcja celu liczyła się możliwie szybko. Lepsza wydajnościowo implementacja pozwoli też na większe zagęszczenie siatki dyskretyzacji i tym samym wzrost dokładności aproksymacji.

Jedną z praktyk pozwalającą poprawić wydajność programów w środowiskach MATLAB i Octave jest tak zwana wektoryzacja. Polega ona na zastępowaniu pętli operacjami na wektorach. Korzysta to z faktu, że wiele funkcji w tych środowiskach można wywołać z wektorem parametrów, zamiast pojedynczego parametru i zwracają one wtedy wektor wyników, oraz wykonują się znacznie szybciej niż gdyby wywołać je wielokrotnie w pętli. Z tego powodu będziemy korzystać z tej techniki gdzie to tylko możliwe.

Aby sprowadzić problem optymalnego sterowania do problemu optymalizacji nieliniowej musimy obliczyć przybliżone sterowanie (5) lub (6), przybliżyć rozwiązanie równania różniczkowego (8) a następnie za pomocą uzyskanego rozwiązania obliczyć kwadraturę (11) przybliżającą funkcję celu (3). Ponadto chcemy obliczyć gradient funkcji celu, więc należy policzyć pochodną każdego z wymienionych równań względem parametrów g_i.

Jak wygląda ∇J ? Stąd wynikanie i czy faktycznie musimy "różniczkować równanie" (?)

2.1 Sterowanie

Jedynym problemem przy implementacji przybliżonego sterowania (5) lub (6) jest znalezienie indeksu i takiego, że $t \in [t_{i-1}, t_i)$. Użyjemy do tego funkcji *lookup*, która wydajnie znajduje żądany indeks korzystając z wyszukiwania binarnego.

Aby policzyć gradient przybliżonego sterowania należy zróżniczkować równanie (5) lub (6):

$$\frac{\partial \hat{g}}{\partial g_i}(t) = \begin{cases} 1 \text{ gdy } t \in [t_{i-1}, t_i) \\ 0 \text{ w.p.p.} \end{cases} \tag{14}$$

lub

$$\frac{\partial \hat{g}}{\partial g_i}(t) = \begin{cases} \frac{t_i - t}{t_i - t_{i-1}} \text{ gdy } t \in [t_{i-1}, t_i) \\ \frac{t - t_i}{t_{i+1} - t_i} \text{ gdy } t \in [t_i, t_{i+1}) \\ 0 \text{ w.p.p.} \end{cases} \tag{15}$$

Implementacja powyższych wzorów jest standardowa, znajdowanie odpowiedniego indeksu i odbywa się tak samo jak przy liczeniu wartości przybliżonego sterowania.

Zarówno funkcja *lookup* jak i pozostałe operacje wykorzystywane do policzenia przybliżonego sterowania i jego gradientu są zwektoryzowane, w szczególności środowiska MATLAB i Octave umożliwiają wektorowe indeksowanie, więc nasza implementacja sterowania i gradientu też jest zwektoryzowana.

2.2 Równanie różniczkowe

Aby przybliżyć rozwiązanie równania (1) należy zaimplementować te równania w postaci funkcji $f(t, y, g)$ gdzie $y = (V_1, V_2, K)^T$, a następnie zaimplementować odpowiednią metodę Rungego-Kutty (8). Wzory te implementuje się bezpośrednio.

Liczenie pochodnych $\frac{\partial \hat{y}}{\partial g_i}$ polega na zróżniczkowaniu wzorów (8):

$$\begin{gathered} \frac{\partial k_1}{\partial g_i} = D_f \cdot \frac{\partial \hat{y}}{\partial g_i}(t_n + ah) + \frac{\partial f}{\partial g_i}(t_n + ah, \hat{y}(t_n + ah), \hat{g}) \\ \frac{\partial k_l}{\partial g_i} = D_f \cdot \left(\frac{\partial \hat{y}}{\partial g_i}(t_n + ah) + h \sum_{j=1}^{l-1} a_{lj} \frac{\partial k_i}{\partial g_j} \right) + \frac{\partial f}{\partial g_i}(t_n + ah, \hat{y}(t_n + ah), \hat{g}) \\ \frac{\partial \hat{y}}{\partial g_i}(t_n + (a+1)h) = \frac{\partial \hat{y}}{\partial g_i}(t_n + ah) + h \sum_{j=1}^{r} b_j \frac{\partial k_j}{\partial g_i} \end{gathered} \tag{16}$$

gdzie

$$D_f = \begin{bmatrix} \frac{\partial f_1}{\partial V_1} & \frac{\partial f_1}{\partial V_2} & \frac{\partial f_1}{\partial K} \\ \frac{\partial f_2}{\partial V_1} & \frac{\partial f_2}{\partial V_2} & \frac{\partial f_2}{\partial K} \\ \frac{\partial f_3}{\partial V_1} & \frac{\partial f_3}{\partial V_2} & \frac{\partial f_3}{\partial K} \end{bmatrix} (t_n + ah, \hat{y}(t_n + ah), \hat{g}) \tag{17}$$

Wzory na elementy macierzy D_f pominiemy, liczy się je i implementuje standardowo.

Nasza implementacja umożliwia też przekazanie wektora punktów w czasie jako argumentu. W takim przypadku w trakcie iteracji będzie spamiętywać wyliczone wartości w tych punktach. Dla wygody założymy, że punkty podane będą w kolejności rosnącej, oraz odległości między sąsiednimi punktami będą podzielne przez długość kroku h.

2.3 Funkcja celu

Przybliżoną funkcję celu (11) można zaimplementować za pomocą iloczynu skalarnego:

$$\hat{J}(\hat{y}) = h_1(\alpha_0, \ldots, \alpha_N) \cdot j(\hat{y}(0, h_1, 2h_1, \ldots, T))^T \tag{18}$$

wydaje się naturalne, żeby $h_1 = h$.

Korzystamy tu ze zwektoryzowanej implementacji funkcji $\hat{y}$, założenia naszej implementacji są spełnione o ile h_1 dzieli h. Implementacja funkcji j też jest zwektoryzowana, ponieważ środowisko dostarcza nam zwektoryzowaną funkcję tanh, a pozostałe operacje wektoryzują się naturalnie.

Gradient funkcji celu możemy zaimplementować jako mnożenie wektora przez macierz:

jak rozumieć ten napis?

$$\frac{\partial J}{\partial(g_1, \ldots g_n)} = h_1(\alpha_0, \ldots, \alpha_N) \cdot \frac{\partial j(\hat{y}(0, h_1, 2h_1, \ldots, T))}{\partial(g_1, \ldots g_n)} \tag{19}$$

gdzie

$$\underline{\frac{\partial j(y(t))}{\partial g_i} = \frac{\partial V_1}{\partial g_i} + \frac{\partial V_2}{\partial g_i} + \omega G'\left(\frac{V_1 - V_2}{\epsilon}\right) \frac{\frac{\partial V_2}{\partial g_i} - \frac{\partial V_1}{\partial g_i}}{\epsilon}, \quad G'(x) = \frac{1}{2\cosh^2(x)}} \tag{20}$$

3 Eksperymenty

Wydaje się, że V_1, V_2 zależą od g?

W eksperymentach użyjemy wartości parametrów podanych w wyjściowej pracy:

$t_0 = 0$	$\alpha_{12} =$??(przyjąłem 0.1)	$\lambda_1 = 0.192$	$\omega =$??(przyjąłem 1000)
$T = 200$	$\alpha_{21} =$??(przyjąłem 0.15)	$\lambda_2 = 0.192$	$\epsilon = 0.01$
$V_{10} = 20$	$\beta_1 = 0.15$	$d = 0.00873$	
$V_{20} = 280$	$\beta_2 = 0.1$	$\mu = 0$	
$K_0 = 650$	$\beta = 0.05$	$g_{\max} = 3$	

Do przybliżania rozwiązania równania (1) będziemy korzystać z metody Rungego-Kutty 4-tego rzędu, ponieważ daje ona dobrą dokładność, a liczenie wartości funkcji f nie jest zbyt kosztowne. Długość kroku obierzemy na $h = 0.1$, co daje 2000 kroków na całym przedziale $[t_0, T]$. Wartości stałych c_l, a_{li}, b_i z wzoru (8) podane na tabeli Butchera:

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array} \tag{21}$$

Do przybliżania funkcjonału celu (3) użyjemy kwadratury trapez~~owej~~ ową oraz długości kroku $h_1 = h = 0.1$. Jest to nieskomplikowana kwadratura, ale przy niskiej długości kroku daje dobre przybliżenie. Wartości parametrów α_i we wzorze (11) to:

$$\alpha_i = \begin{cases} 1 \text{ gdy } i \in \{0, N\} \\ 2 \text{ w.p.p.} \end{cases} \tag{22}$$

Przeprowadzone eksperymenty obejmowały różne punkty startowe dla optymalizatora i siatki dyskretyzacji. Testowane było zarówno sterowanie kawałkami stałe (5), jak i kawałkami liniowe (6). Przykładowymi przetestowanymi punktami startowymi było sterowanie stale równe 0, 1 lub $g_{\max}$, sterowanie malejące liniowo od $g_{\max}$ do 0, czy sterowanie przyjmujące $g_{\max}$ na początku przedziału czasu, i 0 później. Przykładowymi siatkami dyskretyzacji są punkty rozłożone równo w odległości $\frac{1}{2}$, oraz siatka z odstępami $\frac{1}{2}$ w początkowej i końcowej $\frac{1}{4}$ długości przedziału i odstępami 1 na pozostałej częsci przedziału czasu.

Do eksperymentów użyte zostało środowisko Octave. Optymalizator FMINICON na tym środowisku umożliwia użycie jednego z dwóch backendów do optymalizacji nieliniowej: *lm_feasable*, lub *sqp*. Jedną z różnic między tymi backendami jest to, że *lm_feasable* zachowuje ograniczenia w trakcie optymalizacji, natomiast *sqp* wymusza ograniczenia jedynie na koniec procesu optymalizacji. W eksperymentach przetestowane zostały oba z tych algorytmów.

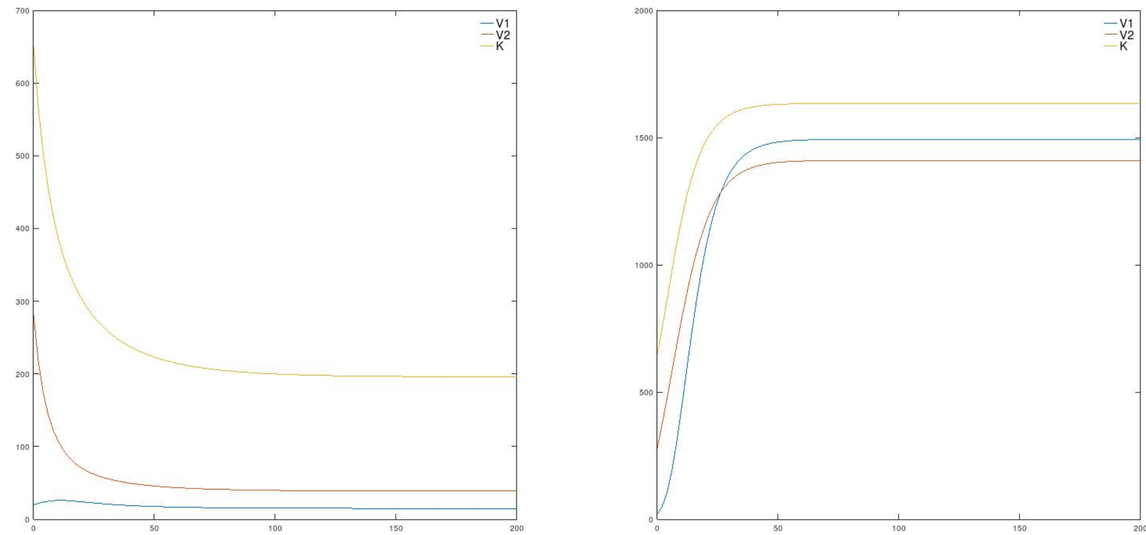
3.1 Wyniki eksperymentów

niepokojące.

Wyniki wszystkich eksperymentów są zgodne: najlepsze rezultaty osiągamy dla sterowania stale równego $g_{\max}$. Optymalizator zaczynając w nim nie znajduje żadnego lepszego punktu, natomiast dla wielu innych punktów startowych (n.p. dla sterowania stałego równego 0 lub 1) zbiega do sterowania niewiele różniącego się od stale równego $g_{\max}$. Stosowanie różnych backendów, siatek optymalizacji oraz metod przybliżania sterowania nie pozwoliło znaleźć istotnie różnego rozwiązania dającego obiecujące wyniki. Żaden eksperyment nie zbiegł do punktu dającego mniejszą wartość przybliżonego funkcjonału celu.

Wartość przybliżonego funkcjonału celu dla sterowania stale równego $g_{\max}$ to ok. $213575,7$. Dla porównania wartość przy braku leczenia, czyli sterowaniu stale równemu 0, to ok. $567688,3$. Przybliżone rozwiązania równania (1) przedstawiono na poniższym wykresie.

(a) $g(t) = g_{\max}$ (b) $g(t) = 0$



Rysunek 1: Rozwiązania równania (1)

Literatura

[1] M. Diehl. *Numerical Optimal Control*. 2011.

Wynik jest odmienny od tego, co pokazywał Pan w referacie.
Co się zmieniło?