AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

**WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI,
INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ**

KATEDRA INFORMATYKI STOSOWANEJ

Praca dyplomowa inżynierska

Biblioteka w językach R/Java wspierająca metodę porównywania parami

R/Java library supporting the pairwise comparisons method

Autor: *Dawid Talaga*
Kierunek studiów: *Informatyka*
Opiekun pracy: *dr har. Konrad Kułakowski*

Kraków, 2017

Uprzedzony o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2006 r. Nr 90, poz. 631 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystycznego wykonania albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.", a także uprzedzony o odpowiedzialności dyscyplinarnej na podstawie art. 211 ust. 1 ustawy z dnia 27 lipca 2005 r. Prawo o szkolnictwie wyższym (t.j. Dz. U. z 2012 r. poz. 572, z późn. zm.): „Za naruszenie przepisów obowiązujących w uczelni oraz za czyny uchybiające godności studenta student ponosi odpowiedzialność dyscyplinarną przed komisją dyscyplinarną albo przed sądem koleżeńskim samorządu studenckiego, zwanym dalej «sądem koleżeńskim».", oświadczam, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i że nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.

Serdecznie dziękuję …...

Spis treści

1. Wprowadzenie

1.1. Metoda porównań parami

Ludzie od wieków podejmują decyzje. Niektóre z nich są proste i przychodzą z łatwością, inne jednak, te bardziej skomplikowane, wymagają głębszej analizy. Jednym z przykładów jest handel wymienny (barter), który opiera się na zamianie określonych towarów. Skąd jednak mieć pewność, że przedmioty mają podobną wartość lub w jaki sposób oszacować koszt określonego towaru? Z takimi problemami musieli mierzyć się nasi pradziadowie. Na szczęście rozwój matematyki przyniósł nam ciekawe narzędzie, metodę porównań parami (ang. The Pairwise Comparisons (PC) method), która w prosty sposób pomaga w udzieleniu odpowiedzi na powyższe pytania. Pierwszym udokumentowanym przypadkiem użycia metody jest binarny system elekcyjny opisany przez Rajmunda Llula (kat. Ramon Llull) już w XIII wieku. Znaczny rozwój metody przypada jednak na wiek XIX i XX i wiąże się z działalnością Fechnera [1] i Thrustone [2]. Przełomowym momentem stało się wprowadzenie w 1980 roku *The Analytic Hierarchy Process (AHP)* [3], dokonane przez Saaty'ego, które pozwoliło na porównywanie coraz to bardziej skomplikowanych obiektów, poprzez tworzenie z nich rozbudowanej hierarchii. Kolejnym krokiem naprzód stało się stworzenie przez doktora Kułakowskiego *Heuristic Rating Estimation (HRE)* [4], które pokazuje nowy kierunek metody oraz nieodkryte dotąd możliwości.

Metoda PC opiera się na założeniu, że zamiast porównywać wszystkie alternatywy od razu, lepiej jest porównać je parami, a następnie zebrać wszystkie wyniki razem [5]. Takie podejście znacznie ułatwia wybór najlepszej alternatywy lub obiektu oraz daje bardziej wiarygodne rezultaty. Stwarza to również możliwość zebrania wyników od wielu osób, np. ekspertów w danej dziedzinie, połączenia ich ocen, porównania poszczególnych wartości i w końcu udzielenia odpowiedzi na pytanie: Która alternatywa (według określonego kryterium) jest naprawdę najlepsza?

1.2. Cele pracy

Celem poniższej pracy jest przedstawienie biblioteki *PariwiseComparisons*, która powstała w ramach pracy inżynierskiej. Biblioteka implementuje 49 funkcji związanych z metodą porównań parami, a jej funkcjonalność pokrywa się z istniejącym już pakietem napisanym w Wolfram Language i dostępnym w Internecie [6].

Biblioteka została stworzona z myślą, aby ułatwić tworzenie aplikacji wykorzystujących metodę porównań parami oraz zachęcić do poznawania jej możliwości. Dzięki tej pracy, programista, który rozwija

taką aplikację, nie będzie musiał skupiać się na szczegółach implementacji i poświęcać wielu godzin na zgłębianie obliczeń matematycznych w danym języku. Celem tworzenia biblioteki było zaimplementowanie funkcji zarówno tych najbardziej podstawowych, np. związanych z tworzeniem macierzy porównań parami, jak również tych bardziej skomplikowanych, potrafiących obliczać całe rankingi. Dlatego pakiet może stać się pomocny nie tylko dla osób zajmujących się na co dzień metodą porównań parami, ale także dla tych, którzy chcą skorzystać tylko z jednej funkcjonalności metody, bez zagłębiania się w szczegóły.

Z powodu dużej popularności języka Java i mnogości aplikacji, które powstają w tym języku, powstała również druga wersja biblioteki, skompresowana do pliku Jar (pariwiseComparisons.jar), gotowa do użycia w projektach pisanych w tym Javie.

1.3. Zawartość i struktura pracy

Na kolejnych kartach niniejszej pracy prezentuję elementy metody porównań parami oraz funkcje z biblioteki PariwiseComPairsons, które implementują poszczególne funkcjonalności. W treści zawarte są również podpowiedzi w jaki sposób użyć pakietu w języku R oraz jak łatwo wywoływać funkcje z poziomu maszyny wirtualnej Javy.

Szczególną uwagę należy zwrócić na specyficzną strukturę pracy. Główna część, prezentująca metodę porównań parami, została podzielony na kilka mniejszych rozdziałów (2-5). W każdym z nich pierwszą część stanowią rozważania teoretyczne, bazujące na pracy naukowców i ich odkryciach. Druga część to prezentacja funkcji biblioteki PairwiseComPairsons, ze szczegółowym omówieniem najważniejszych z nich. Mam nadzieję, że taka budowa tej pracy pozwoli czytelnikowi poznać podstawowe zagadnienia związane z metodą porównań parami oraz zapoznać się z konkretnymi funkcjami przeznaczonymi do poszczególnych zagadnień.

2. Ranking AHP

2.1. Macierz PC

Kiedy chcemy podjąć prostą decyzję, sytuacja często sprowadza się do porównania dwóch wartości, np. gdy naszym zadaniem jest wybór cięższego owocu, sprawa jest oczywista. Wystarczy zważyć owoce lub oszacować ich wagę, a następnie wybrać cięższy. Wiele decyzji, z którymi spotykamy się na co dzień, jest jednak o wiele bardziej skomplikowanych. Może tak być w przypadku, gdy naszego kryterium nie można łatwo zmierzyć (np. użyteczność lub atrakcyjność), porównywanych obiektów jest dużo i znacznie się od siebie różnią lub są złożone, a kryterium wyboru zależy od kilku czynników. W każdej z tych sytuacji możemy posłużyć się metodą porównań parami, a konkretnie rankingiem AHP.

W celu utworzenia rankingu na samym początku należy rozpoznać problem, to znaczy zdefiniować decyzję, którą chcemy podjąć oraz znaleźć wszystkie opcje (alternatywy) brane pod uwagę. Następnie tworzymy kwadratową macierz *(PC matrix)*, w której nagłówkami kolumn i wierszy są zdefiniowane przez nas alternatywy.

Kolejnym krokiem jest uzupełnienie macierzy poprzez wykonanie odpowiedniej ilości porównań. Każdą alternatywę porównujemy z wszystkimi pozostałymi, tak więc dla n alternatyw otrzymujemy $\frac{n(n-1)}{2}$ porównań. Wyniki wpisujemy do macierzy, na przecięciu odpowiadających opcji.

Istnieje kilka skal, które możemy wykorzystać w czasie porównań, najbardziej popularną jest jednak zaproponowana przez Saaty'ege *(Satty's scale)* [7], którą również ja posłużę się w przykładach. Skala Saaty'ego przyjmuje następujące wartości:

$$\{\frac{1}{9}, \frac{1}{8}, \ldots, \frac{1}{2}, 1, 2, \ldots, 8, 9\}$$

Ogólnie można powiedzieć, że im wyższa wartość przydzielona dla danej alternatywy, tym lepiej wypada ona w stosunku do drugiej opcji, a więc wartość $\frac{1}{9}$ można tłumaczyć jako *ekstrelamnie nie preferuję*, $\frac{1}{3}$ jako *nie preferuję*, 1 jako *tak samo ważna jak druga*, a 5 jako *silnie preferuję*, itd. Oczywiście nie porównujemy ze sobą tych samych alternatyw, lecz od razu wpisujemy do macierzy wartość 1.
Etap porównywania alternatyw i wyboru numerycznej wartości tego porównania jest najsłabszą stroną metody porównań parami, gdyż zależy od czynnika ludzkiego i narażony jest na błędy oraz niespójności. Problemy z tego wynikające opisuję w rozdziale (4).

Po wykonaniu wszystkich porównań i wpisaniu wartości otrzymujemy kompletną macierz PC, którą formalnie zapisujemy jako:

$$M = (m_{ij}) \wedge m_{i,j} \in R_+ \wedge i, j \in \{1, \ldots, n\}$$

Natomiast zbiór alternatyw oznaczamy jako

$$X = \{x_1, \ldots, x_n\}$$

Macierz dla n alternatyw wygląda więc następująco:

$$M = \begin{pmatrix} 1 & m_{12} & \ldots & m_{1n} \\ m_{21} & 1 & \ldots & m_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ m_{n1} & m_{n2} & \ldots & 1 \end{pmatrix}$$

Warto zauważyć, że wartości wykonanych przez nas porównań wpisujemy nad przekątną (tworząc *Upper Triangular Matrix*, natomiast pod nią automatycznie wprowadzamy wartości odwrotne. Nielogiczne byłoby ponowne porównywanie tych samych alternatyw, z odwróconą jedynie kolejnością. Ogólnie można powiedzieć, że zachodzi warunek $a_{ji} = \frac{1}{a_{ij}}$. Na przekątnej zaś zawsze znajdą się same jedynki.

Przyklad 2.1.1. *Chcemy podjąć decyzję, gdzie pojechać na wakacje. Rozważamy trzy opcje: Gdańsk, Zakopane lub Barcelona. W takim przypadku nasz zbiór alternatyw wygląda bardzo prosto:* $X = \{Gdansk, Zakopane, Barcelona\}$. *Następnie dokonujemy porównań parami, z których wynika, że bardziej preferujemy wyjazd do Gdańska niż do Zakopanego w stosunku* $3/1$*, parę Gdańsk - Barcelona oceniamy na* $1/2$*. Natomiast z dwójki Zakopane i Barcelona wybieramy Barcelonę i przypisujemy jej wartość* 6 *(silnie preferowana). W takim przypadku macierz porównań parowych wygląda następująco:*

$$M = \begin{pmatrix} 1 & 3 & \frac{1}{2} \\ \frac{1}{3} & 1 & \frac{1}{6} \\ 2 & 6 & 1 \end{pmatrix}$$

2.2. Wektor wag

Kolejnym krokiem do podjęcia decyzji jest utworzenie wektora wag (nazywany również *wektorem priorytetów*). Służy do tego funkcja, która każdej alternatywie ze zbioru X przyporządkowuje dodatnią liczbę rzeczywistą. Do wyliczenia wektora wykorzystuje macierz PC.

$$w = [w(c_1), \ldots, w(c_n)]^T$$

Wysoka wartość w wektorze wag oznacza, że dana alternatywa jest mocno preferowana. Bez trudu dochodzimy więc do wniosku, że najlepszą opcją w danym problemie jest ta, dla której przypisano najwyższą wartość w wektorze wag.

Dość naturalnie pojawia się pytanie, w jaki sposób działa funkcja wyznaczająca wektor wag. Istnieje kilka sposobów wyliczania tych wartości, poniżej przedstawię dwa najbardziej popularne, stosunkowo proste i dające rzetelne rezultaty:

1. Na podstawie macierzy PC wyznaczony zostaje wektor własny korespondujący z najwyższą wartością własną tej macierzy.

2. Dla każdego wiersza macierzy PC wyliczona zostaje średnia geometryczna wartości w danym wierszu.

Warto zaznaczyć, że otrzymany wektor wag należy przeskalować, w taki sposób, aby suma wartości w nim zawartych wynosiła 1.

Przyklad 2.2.1. *Na podstawie wyznaczonej w poprzenim przykładzie macierzy PC, korzystając z metody opartej na wektorach własnych, wyznaczamy wektor wag:*

$$w = \begin{bmatrix} 0.4423259 & 0.1474420 & 0.8846517 \end{bmatrix}^T$$

Po przeskalowaniu otrzymujemy kompletny wektor wag:

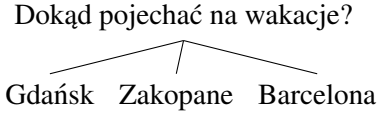
$$w = \begin{bmatrix} 0.3 \\ 0.1 \\ 0.6 \end{bmatrix}$$

Z wyznaczonego wektora wag możemy wyciągnąć wnioski, że najlepszą alternatywą na wakacyjny wyjazd jest Barcelona, a najmniej preferowaną Zakopane. W ten prosty sposób udało nam się udzielić odpowiedzi na zadane pytanie.

2.3. AHP

Zdarzają się jednak sytuacje, w których problem jest bardziej skomplikowany, a alternatyw nie można zapisać w jednej macierzy porównań parowych. Wtedy z pomocą przychodzi wprowadzone przez Saaty'ego *The Analytic Hierarchy Process (AHP)* [3].

Podstawą AHP jest budowa hierarchii, która obrazuje proces powstawania decyzji. Prezentuje ona kolejne poziomy, od najbardziej ogólnego (decyzja), do szczegółowych, które mówią jakie porównania należy wykonać, aby rozwiązać problem. Najprostszą hierarchię mogliśmy zbudować już w poprzednim przykładzie, wyglądałaby ona następująco:

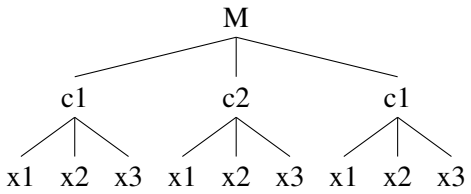


Nie mówiliśmy wtedy jednak o AHP, gdyż wybór opierał się tylko na jednej macierzy PC.

AHP wprowadza kolejne pojęcie - kryterium. Kryterium to cecha posiadana przez każdą alternatywę i która wywiera wpływ na podejmowane decyzje. W celu stworzenia AHP i rozwiązania złożonego problemu należy zdefiniować zbiór kryteriów C.

$$C = \{c_1, \ldots, c_m\}$$

Następnym krokiem jest określenie kryteriów dla każdej alternatywy. Otrzymujemy wtedy kompletne dane, które pozwalają nam zbudować następującą hierarchię AHP:

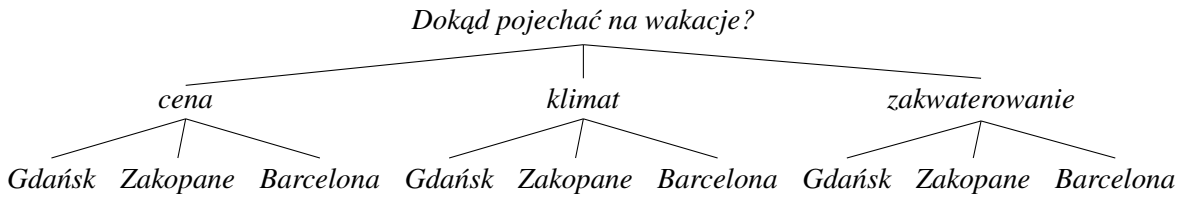


Przyklad 2.3.1. *Ponownie naszym zadaniem jest wybór miejsca, w które chcemy pojechać na wakacje. Oprócz zbioru alternatyw $X = \{Gdansk, Zakopane, Barcelona\}$, tym razem podajemy także kryteria, które są dla nas ważne w czasie podejmowania wyboru: $C = \{cena, klimat, zakwaterowanie\}$.*
Dla każdej z alternatyw określamy jej szczegóły:

Tabela 2.1. Szczegóły każdej alternatywy

	Alternatywy		
Kryteria	Gdańsk	Zakopane	Barcelona
cena	800zł	1300zł	2400zł
klimat	górski	bałtycki	śródmorski
zakwaterowanie	namiot	pensjonat	hotel

Na podstawie powyższych danych budujemy drzewo AHP:



Po utworzeniu hierarchii AHP możemy przystąpić do budowy macierzy PC. Tym razem nie będzie to jedna tablica, lecz kilka, które powiążemy ze sobą. Dla każdego poddrzewa składającego się z jednego wierzchołka-rodzica i kilku wierzchołków-dzieci konstruujemy osobną macierz PC oraz wykonujemy

odpowiednie porównania. Następnie wyznaczamy przeskalowany wektor wag dla każdej macierzy. Jedyną różnicą w porównaniu z pierwszym przykładem jest sposób wyniku. Aby otrzymać odpowiedni wektor wynikowy, którego element sumują się do 1, należy zsumować wektory powstałe z macierzy PC na tym samym poziomie (najgłębszy poziom struktury), przemnażając je przez odpowiadające im wartości wag z poziomu wyższego. Tworzymy w ten sposób kombinację liniową wektorów priorytetów [8]. Dla n alternatyw wektor wag obliczamy według wzoru:

$$w = \hat{w}_1 w^{(c1)} + \hat{w}_2 w^{(c2)} + \cdots + \hat{w}_n w^{(c3)},$$

gdzie
$\hat{w}_1, \hat{w}_2, \ldots$ to kolejne elementy wektora wag kryteriów,
$w^{(c1)}, w^{(c2)}, \ldots$ to kolejne wektory wag obliczone dla poszczególnych kryteriów

Może zdarzyć się sytuacja, w której rozważany problem jest jeszcze bardziej rozbudowany. Wtedy powyższe kroki należy powtórzyć na każdym poziomie zagłębienia hierarchii AHP.

Przyklad 2.3.2. *W naszym przykładzie należy zbudować cztery macierze PC: jedna zdefiniuje priorytety kryteriów, trzy pozostałe określą preferencje w ramach poszczególnych kryteriów. Dla każdej macierzy wykonujemy odpowiednie porównania, a wyniki wpisujemy do macierzy:*

$$\hat{M} = \begin{pmatrix} 1 & \frac{1}{2} & 3 \\ 2 & 1 & 4 \\ \frac{1}{3} & \frac{1}{4} & 1 \end{pmatrix}$$

$$M_{cena} = \begin{pmatrix} 1 & 3 & 7 \\ \frac{1}{3} & 1 & 4 \\ \frac{1}{7} & \frac{1}{4} & 1 \end{pmatrix} \qquad M_{klimat} = \begin{pmatrix} 1 & 1 & \frac{1}{2} \\ 1 & 1 & \frac{1}{2} \\ 2 & 2 & 1 \end{pmatrix} \qquad M_{zakwaterowanie} = \begin{pmatrix} 1 & 2 & \frac{1}{3} \\ \frac{1}{2} & 1 & \frac{1}{4} \\ 3 & 4 & 1 \end{pmatrix}$$

Wyznaczamy wektor wag dla każdej macierzy

$$\hat{w} = \begin{bmatrix} 0.32 \\ 0.56 \\ 0.12 \end{bmatrix} \qquad w_c = \begin{bmatrix} 0.66 \\ 0.26 \\ 0.08 \end{bmatrix} \qquad w_k = \begin{bmatrix} 0.25 \\ 0.25 \\ 0.50 \end{bmatrix} \qquad w_z = \begin{bmatrix} 0.24 \\ 0.14 \\ 0.62 \end{bmatrix}$$

Warto zwrócić uwagę, że wektor oznaczony jako $\hat{w}$ jest poziom wyżej od pozostałych i przez jego wartości będziemy przemnażać pozostałe wektory.

Ostateczny wynik:

$$w = 0.32 w_c + 0.56 w_k + 0.12 m u_z = \begin{bmatrix} 0.379 \\ 0.240 \\ 0.381 \end{bmatrix}$$

Otrzymaliśmy rezultat i rozwiązanie zdefiniowanego problemu. Najlepszym miejscem na wakacyjny wyjazd okazała się Barcelona. Osiągnęła jednak minimalną przewagę nad Gdańskiem.

2.4. Biblioteka PairwiseComparisons - AHP

Przedstawię kilka zasad tworzenia zmiennych w językach R i Java, które można przekazywać do funkcji z biblioteki PairwiseComparisons.

Informacje dotyczące języka R:

1. Aby utworzyć wektor wystarczy wywołać funckję *c*, przekazując jej dowolną ilość wartości.
2. Do transpozycji wektora służy funkcja *t*.
3. Istnieje kilka funkcji, które tworzą macierz, najważniejsze z nich to:
   - *matrix* – jako argumenty przekazujemy wektor z danymi, liczbę wierszy i liczbę kolumn.
   - *rbind* – jako argumenty przekazujemy wektory z danymi. Każdy z wektorów utworzy jeden wiersz.
   - *cbind* – jako argumenty przekazujemy wektory z danymi. Każdy z wektorów utworzy jedną kolumną.

   Dodatkowe opcje każdej z funkcji można doczytać w dokumentacji. Przykładowe wywołania: *matrix(c(1,2,0.5,1),2,2)*; *rbind(c(1,2),c(0.5,1))*; *cbind(c(1,0.5),c(2,1))*;

Informacje dotyczące języka Java:

1. Aby przekazać macierz do funkcji z omawianej biblioteki, wystarczy utworzyć i uzupełnić danymi tablicę dwuwymiarową typu *double*.
2. Aby przekazać wektor do funkcji, należy utworzyć i zainicjalizować tablicę typu *double*.

Przykładowe utworzenie odpowiednich zmiennych:
double[2][2] matrix = {{1,0.5},{2,1}};
double[2] vector = {0.4,0.6};

Dla niektórych funkcji powstały również wersje symboliczne, które mają dokładnie takie same argumenty, zwracają tę samą wartość i działają minimalnie szybciej. W wyjątkowych sytuacjach może jednak się zdarzyć, że wersja symboliczna nie zwróci prawidłowego rezultatu.

Funkcje z biblioteki PairwiseComparisons pomocne w powyższych obliczeniach:

Największa wartość własna macierzy

Nazwa
principalEigenValue

Opis
Oblicza największą wartość własną z macierzy.

Argumenty
matrix – PC matrix

Zwracana wartość
Największa wartość własna macierzy

Dodatkowe informacje
W bibliotece dostępna jest również symboliczna wersja tej funkcji *principalEigenValueSym*

Wektor własny macierzy

Nazwa
principalEigenVector

Opis
Oblicza wektor własny macierzy korespondujący z jej największą wartością własną (patrz *principalEigenValue*).

Argumenty
matrix – PC matrix

Zwracana wartość
Wektor własny macierzy

Dodatkowe informacje
W bibliotece dostępna jest również symboliczna wersja tej funkcji *principalEigenVectorSym*

Ranking na podstawie wartości własnych

Nazwa
eigenValueRank

Opis
Oblicza ranking macierzy na podstawie metody wartości własnych. Oblicza wektor własny, a następnie przekształca go w taki sposób, że suma elementów wynosi 1.

Argumenty
matrix – PC matrix

Zwracana wartość
Przeskalowany wektor własny macierzy

Dodatkowe informacje
W bibliotece dostępna jest również symboliczna wersja tej funkcji *eigenValueRankSym*

Wektor średnich geometrycznych

Nazwa

geometricRank

Opis

Oblicza wektor, którego elementy są średnimi geometrycznymi każdego wiersza przekazanej macierzy. Wektor ten posłuży do stworzenia rankingu opartego o metodę średnich geometrycznych.

Argumenty

matrix – PC matrix

Zwracana wartość

Wektor średnich geometrycznych

Ranking na podstawie średnich geometrycznych

Nazwa

geometricRescaledRank

Opis

Oblicza ranking macierzy na podstawie metody średnich geometrycznych. Oblicza wektor, którego elementy są średnimi geometrycznymi każdego wiersza przekazanej macierzy, a następnie skaluje ten wektor w taki sposób, że suma elementów wynosi 1.

Argumenty

matrix – PC matrix

Zwracana wartość

Przeskalowany wektor średnich geometrycznych macierzy

Ranking AHP na podstawie wartości własnych

Nazwa

ahp

Opis

Oblicza wielokryteriowy ranking AHP na podstawie metody bazującej na wartościach własnych macierzy. Wykorzystuje macierz kryteriów i macierze alternatyw. Jest to podstawowy, trójpoziomowy ranking AHP. Ilość przekazanych wartości do funkcji zależy od użytkownika i wynosi $n + 1$, gdzie n to ilość kryteriów w rozważanym problemie.

Argumenty

M – macierz kryteriów o wymiarach *nxn*
... – lista n macierzy PC, z których każda dotyczy jednego kryterium.

Zwracana wartość

Ranking AHP

3. Heuristic Rating Estimation

3.1. Wstęp do HRE

Istnieją również inne decyzje i problemy do rozwiązania. To sytuacje, w których pewne wartości są niepodważalne, z góry określone lub narzucone przez kogoś. Nie chcemy nimi manipulować ani dyskutować z ich wiarygodnością. Zależy nam natomiast, aby do tej relacji wprowadzić nowe obiekty, których wartość (w szerokim tego słowa znaczeniu) określimy w tej samej skali. Jako przykład możemy wyobrazić sobie targ z owocami, na którym sadownicy wymieniają się zbiorami bez użycia pieniędzy. Jedyną ustaloną i tradycyjną już wymianą jest sprzedawanie dwóch gruszek w zamian za trzy jabłka. Trzymając się tego wyznacznika, chcemy określić *ceny* innych owoców. Jak tego dokonać?

W takiej sytuacji z pomocą przychodzi nam wprowadzone przez doktora Kułakowskiego *Heuristic Rating Estimation (HRE)*. W mojej pracy ogólnie omówię to zagadnienie, zainteresowanych szczegółami odsyłam do źródła [4] [9].

3.2. Zbiór alternatyw

Pierwszym ważnym elementem HRE jest zbiór alternatyw. W tej metodzie nie będzie on już tylko zbiorem cech lub obiektów, których wartości poszukujemy. HRE zakłada, że pewne wartości są z góry określone. Nie chcemy ich zmieniać, w szczególności manipulować ich wzajemną relacją. Waga tych obiektów pozostanie niezmienna w czasie obliczeń. Drugą część zbioru tworzą jednak alternatywy, których wartości poszukujemy. Interesuje nas ich stosunek względem siebie i względem wag, które znamy. Jeśli więc poprzez C_K oznaczymy alternatywy, których wartości są znane od początku (*known concepts*), a poprzez C_U opcje, których wartości chcemy przybliżyć (*unknown concepts*), to zbiór alternatyw określamy jako sumę wszystkich elementów i zapisujemy jako:

$$C = C_K \cup C_U$$

3.3. Macierz PC w metodzie HRE

Drugim krokiem, podobnie jak w AHP, jest stworzenie macierzy porównań parowych, która posłuży do dalszych obliczeń. Przykładowa macierz PC wygląda następująco:

$$M = \begin{pmatrix} 1 & m_{12} & m_{13} & m_{14} \\ m_{21} & 1 & m_{22} & m_{24} \\ m_{31} & m_{32} & 1 & \frac{2}{3} \\ m_{41} & m_{42} & \frac{3}{2} & 1 \end{pmatrix}$$

Przedstawiona macierz obrazuje sytuację, w której znamy wartości trzeciej i czwartej alternatywy, łatwo więc obliczamy ich stosunek, który wpisujemy bezpośrednio do macierzy. Z tak przygotowanej macierzy generujemy porównania parowe, które należy wykonać, a wyniki wpisujemy w odpowiednie miejsca. Porównań, w zależności od sytuacji, możemy dokonać sami lub poprosić o nie ekspertów w danej dziedzinie. Oczywiście pomijamy te, które są już znane, a więc w naszym przypadku nie będziemy szacować stosunku alternatywy trzeciej do czwartej.

3.4. Metody HRE

Kolejnym i zarazem najważniejszym krokiem omawianego problemu są obliczenia, które należy wykonać na uzupełnionej macierzy PC. Z racji na ich stopień zaawansowania oraz fakt, że bardzo czytelnie i szczegółowo zostały przedstawione w źródle, nie będę w pełni ich przytaczał. Pokażę tylko kilka etapów składających się na tę metodę i do których odpowiednie funkcje zostały zaimplementowane w bibliotece.

Głównym zagadnieniem jest sprowadzenie problemu to równania postaci

$$Aw = b,$$

gdzie:
A to macierz $r \times r$, gdzie r to ilość elementów poszukiwanych, oznaczamy $|C_U|$,
b to wektor wartości wyliczonych na podstawie znanych alternatyw,
w to poszukiwany wektor wag postaci

$$w = \begin{bmatrix} w(c_1) \\ \vdots \\ w(c_U) \end{bmatrix}$$

Po wyliczeniu przedstawionego równania, wektor w uzupełniamy o wartości ze zbioru C_K i otrzymujemy ostateczny rezultat. Należy pamiętać, że suma elementów w wyliczonym wektorze w jest różna od 1, gdyż wartości znane nie zmieniły się, alternatywy wyliczone zaś są również podane w odniesieniu do nich. Jeśli zależy nam, aby suma elementów wyniosła 1, możemy przeskalować wektor, dzieląc każdą wartość przez sumę wszystkich elementów.

Przyklad 3.4.1. *Prowadzimy handel wymienny owoców. Wiemy, że gruszka jest* 1.5 *razy cenniejsza od jabłka. W zbiorach mamy jeszcze brzoskwienie, truskawki i maliny. Chcemy oszacować wartości wszystkich owoców. Sporządzamy więc zbiory alternatyw znanych, nieznanych i wszystkich razem.*

$$C_K = [jabko, gruszka], \qquad C_U = [malina, brzoskwinia, truskawka].$$
$$C = [jabko, gruszka, malina, brzoskwinia, truskawka]$$

Budujemy macierz PC. Przeprowadzamy odpowiednie porównania parami, a wyniki zapisujemy do macierzy.:

$$M = \begin{pmatrix} 1 & \frac{2}{3} & 10 & 4 & 7 \\ \frac{3}{2} & 1 & 15 & 3 & 5 \\ \frac{1}{10} & \frac{1}{15} & 1 & \frac{1}{3} & \frac{1}{2}\frac{1}{4} \\ \frac{1}{3} & 3 & 1 & 2 & \\ \frac{1}{7} & \frac{1}{5} & 2 & \frac{1}{2} & 1 \end{pmatrix}$$

Korzystając z funkcji HRE możemy od razu wyliczyć wektor wag, aby zaprezentować jednak sposób działania metody, prześledźmy kolejne etapy powstawania wyniku.
Wykorzystujemy algorytmy do obliczenia macierzy A, wektora b, a następnie w:

$$Ab = w$$

$$A = \begin{pmatrix} 1 & -\frac{1}{12} & -\frac{1}{8} \\ -\frac{3}{4} & 1 & -\frac{1}{2} \\ -\frac{1}{2} & -\frac{1}{8} & 1 \end{pmatrix} \qquad b = \begin{bmatrix} 0.1 \\ 0.375 \\ 0.2214286 \end{bmatrix}$$

Po rozwiązaniu otrzymujemy:

$$w = \begin{bmatrix} 0.22 \\ 0.75 \\ 0.42 \end{bmatrix}.$$

Dodajemy wartości znane i otrzymujemy ostateczny rezultat:

$$\hat{w} = \begin{bmatrix} 2 \\ 3 \\ 0.2150966 \\ 0.7475316 \\ 0.4224183 \end{bmatrix}.$$

Ostatnią rzeczą, na którą należy zwrócić uwagę jest kwestia wyboru metod, które służą do obliczeń. W Heuristic Rating Estimation, podobnie jak w AHP możemy wybrać sposób liczenia oparty na wartościach własnych macierzy lub średnich geometrycznych. Obie drogi dają bardzo zbliżone rezultaty. W przykładzie posłużyłem się metodą z wykorzystaniem wartości własnych.

3.5. Biblioteka PairwiseComparisons - HRE

Funkcje z biblioteki PairwiseComparisons pomocne w powyższych obliczeniach:

Macierz A metody HRE

Nazwa
HREmatrix

Opis
W oparciu o wartości własne macierzy, oblicza macierz, która razem z wektorem b (patrz *HREconstantTermVector*) utworzy układ równań liniowych postaci $Aw = b$.

Argumenty
matrix – PC matrix
knownVector – wektor znanych alternatyw, pozostałe oznaczone jako 0

Zwracana wartość
Macierz A, służąca do utworzenia równania $Aw = b$.

Wektor b znanych wartości metody HRE (wykorzystuje wartości własne)

Nazwa

HREconstantTermVector

Opis

Na podstawie znanych alternatyw, w oparcu o wartości własne macierzy, oblicza wektor b, który razem z macierzą A (patrz *HREmatrix*) utworzy układ równań liniowych postaci $Aw = b$.

Argumenty

matrix – PC matrix

knownVector – wektor znanych alternatyw, pozostałe oznaczone jako 0

Zwracana wartość

Wektor b, służący do utworzenia równania $Aw = b$.

Ranking nieznanych wartości HRE (wykorzystuje wartości własne)

Nazwa

HREpartialRank

Opis

W oparcu o wartości własne macierzy oblicza nieznane alternatywy.

Argumenty

matrix – PC matrix

knownVector – wektor znanych alternatyw, pozostałe oznaczone jako 0

Zwracana wartość

Wartości nieznanych alternatyw

Pełny ranking HRE (wykorzystuje wartości własne)

Nazwa

HREfullRank

Opis

W oparciu o wartości własne macierzy oblicza nieznane alternatywy i dodaje je do wektora znanych wartości.

Argumenty

matrix – PC matrix

knownVector – wektor znanych alternatyw, pozostałe oznaczone jako 0

Zwracana wartość

Wektor wag znanych i nieznaych alternatyw.

Przeskalowany ranking HRE (wykorzystuje wartości własne)

Nazwa

HRErescaledRank

Opis

Oblicza pełny ranking HRE (patrz *HREfullRank*), a następnie skaluje wynikowy wektor wag w taki sposób, aby suma elementów wyniosiła 1.

Argumenty

matrix – PC matrix

knownVector – wektor znanych alternatyw, pozostałe oznaczone jako 0

Zwracana wartość

Przeskalowany wektor wag znanych i nieznanych alternatyw.

Macierz A metody HRE (wykorzystuje średnie geometryczne)

Nazwa

HREgeomMatrix

Opis

W oparciu o średnie geometryczne wierszy macierzy, oblicza macierz, która razem z wektorem b (patrz *HREgeomConstantTermVector*) utworzy układ równań liniowych postaci $Aw = b$.

Argumenty

matrix – PC matrix

knownVector – wektor znanych alternatyw, pozostałe oznaczone jako 0

Zwracana wartość

Macierz A, służąca do utworzenia równania $Aw = b$.

Wektor b znanych wartości metody HRE (wykorzystuje średnie geometryczne)

Nazwa

HREgeomConstantTermVector

Opis

Na podstawie znanych alternatyw, w oparciu o średnie geometryczne, oblicza wektor b, który razem z macierzą A (patrz *HREgeomMatrix*) utworzy układ równań liniowych postaci $Aw = b$.

Argumenty

matrix – PC matrix

knownVector – wektor znanych alternatyw, pozostałe oznaczone jako 0

Zwracana wartość

Wektor b, służący do utworzenia równania $Aw = b$.

Pośredni ranking nieznanych wartości HRE

Nazwa

HREgeomIntermediateRank

Opis

W oparciu o średnie geometryczne oblicza podstawę, która posłuży do obliczenia wartości nieznanych alternatyw. Współczynniki te zostaną przemnożone przez 10 (patrz *HREgeomPartialRank*)

Argumenty

matrix – PC matrix

knownVector – wektor znanych alternatyw, pozostałe oznaczone jako 0

Zwracana wartość

Wektor pośrednich wartości nieznanych alternatyw

Ranking nieznanych wartości HRE (wykorzystuje średnie geometryczne)

Nazwa

HREgeomPartialRank

Opis

W oparciu o średnie geometryczne oblicza nieznane alternatywy.

Argumenty

matrix – PC matrix

knownVector – wektor znanych alternatyw, pozostałe oznaczone jako 0

Zwracana wartość

Wartości nieznanych alternatyw

Pełny ranking HRE (wykorzystuje średnie geometryczne)

Nazwa

HREgeomFullRank

Opis

W oparciu o średnie geometryczne oblicza nieznane alternatywy i dodaje je do wektora znanych wartości.

Argumenty

matrix – PC matrix

knownVector – wektor znanych alternatyw, pozostałe oznaczone jako 0

Zwracana wartość

Wektor wag znanych i nieznaych alternatyw.

Przeskalowany ranking HRE (wykorzystuje średnie geometryczne)

Nazwa

HREgeomRescaledRank

Opis

Oblicza pełny ranking HRE (patrz *HREgeomFullRank*), a następnie skaluje wynikowy wektor wag w taki sposób, aby suma elementów wyniosiła 1.

Argumenty

matrix – PC matrix

knownVector – wektor znanych alternatyw, pozostałe oznaczone jako 0

Zwracana wartość

Przeskalowany wektor wag znanych i nieznanych alternatyw.

4. Niespójność

4.1. Problemy związane z metodami porównań parami

Głównym problemem metody porównań parami jest niespójność danych. To najczęstszy zarzut, jaki można usłyszeć ze strony krytyków. Być może to także jeden z powodów, dla których AHP i HRE nie zdobyły jeszcze tak dużej popularności. Mimo, iż metoda opiera się na obliczeniach matematycznych i jest potwierdzona dowodami oraz twierdzeniami, zawiera jednak jeden *słabszy* element - czynnik ludzki. Człowiek współtworzy przecież obliczenia tej metody poprzez dostarczanie wyników porównań, bez których pozostałe składniki nie mają sensu. Dlaczego tak trudno jest dostarczyć spójne dane wejściowe i czy eksperci, którzy proszeni są o dokonanie porównań nie mogliby się tego po prostu nauczyć?

Oczywiście ludzie, którzy znają zasady działania metody potrafią tak dobrać wartości macierzy PC, aby była ona spójna. Jeśli jednak widzimy tylko sparowane alternatywy i mamy przypisać im, zgodnie z naszymi odczuciami, preferencje, okazuje się to już o wiele trudniejsze. Czasem zdarzają się sytuacje, w których ciężko jest przypisać konkretne liczby do stosunku, jaki posiadamy do przedstawionych alternatyw lub określić *stopień preferowania*. Skale ocen spośród których wybieramy ocenę jest umowna, a konkretne wartości mogą zostać różnie, subiektywnie odczytane przez poszczególne osoby.

Zastanówmy się kiedy macierz jest niespójna. Aby łatwiej zrozumieć na czym polega problem, przedstawię dwa proste przykłady. Pierwszy z nich okaże się niespójny nawet bez zwracania uwagi na konkretne wartości.

Przyklad 4.1.1. *Sporządzamy ranking zabawek, aby wybrać ulubiony przedmiot dziecka. Przedstawiamy mu piłkę i rower, a dziecko wybiera piłkę. Następnie pokazujemy rower i hulajnogę, wybór pada na rower. Ostatnie porównanie to hulajnoga i piłka, tym razem dziecko wskazuje na hulajnogę.*

Nie trzeba znać się na matematyce, aby szybko zorientować się, że ranking w tej sytuacji nie ma sensu, ponieważ w teorii, jeśli obiekt A jest lepszy od B, zaś B lepszy od C, to oczekujemy, że obiekt A jest zdecydowanie bardziej preferowany niż C. W praktyce czasami okazuje się inaczej. W tej sytuacji dane są niespójne.

Przyklad 4.1.2. *Porównujemy obiekt A z obiektem B i przypisujemy rezultat 2. Następnie zestawiamy B i C, tutaj również wybieramy wartość 2. A więc, mówiąc potocznie, obiekt A jest dwa razy lepszy od B, który z kolei 2 razy lepszy od C. Jakiego rezultatu oczekujemy więc w porównaniu obiektów A i C?*

Po chwili namysłu dochodzimy do wniosku, że obiekt A w stosunku do obiektu C powinien przyjąć wartość 4. Właśnie wtedy nasze dane będą całkowicie spójne.

Drugi z przedstawionych przykładów prowadzi nas do wniosku, na którym opiera się teoria spójności danych w metodzie porównań parami:

$$m_{ik} = m_{ij}m_{jk} \qquad \forall_{i,j,k} \tag{4.1}$$

W praktyce okazuje się, że bardzo rzadko otrzymujemy idealnie spójną macierz, dlatego do metod porównań parami wprowadzony został współczynnik niespójności. Informuje on o stopniu niespójności danych i na jego podstawie możemy zdecydować, czy warto wykonywać obliczenia na danej macierzy PC, czy może należy poprosić o ponowne wykonanie porównań. Na przestrzeni lat powstało wiele sposobów obliczania współczynnika niespójności, w mojej pracy przedstawię dwa z nich.

4.2. Współczynnik Saaty'ego

Aby wyznaczyć współczynnik Saaty'ego należy ponownie wykorzystać maksymalną wartość własną macierzy. To właśnie w oparciu o ten parametr Saaty przedstawił swoje rozważania [7]. Wykorzystał fakt, że największa wartość własna każdej macierzy jest równa jej wymiarowi wtedy i tylko wtedy, kiedy dana macierz jest spójna. Na tej podstawie zaproponował współczynnik niespójności (ang. *Consistency Index*). Dla macierzy o wymiarze *n* wyraża się wzorem:

$$CI(A) = \frac{\lambda_{max} - n}{n-1}$$

W najprostszej wersji obliczania współczynnika niespójności możemy w tym miejscu zakończyć nasze rozważania. Przyjmuje się, że jeżeli wyznaczona wartość CI jest mniejsza niż 0.1, to macierz jest spójna, w przeciwnym wypadku należy poprawić wartości porównań.

Przyklad 4.2.1.

$$M = \begin{pmatrix} 1 & 2 & 8 \\ \frac{1}{2} & 1 & \frac{3}{4} \\ \frac{1}{8} & \frac{4}{3} & 1 \end{pmatrix}$$

Wyznaczamy największą wartość własną: $\lambda_{max} = 3.319518$*, a następnie współczynnik:*

$$CI(M) = \frac{3.319518 - 3}{3-1} = 0.159759.$$

Otrzymany rezultat informuje nas, że macierz M *nie jest spójna.*

Nieco bardziej dokładny sposób obliczania niespójności zaproponowany przez Saaty'ego zestawia wartość CI z współczynnikiem zależnym od wymiaru macierzy. Pozwala to na bardziej szczegółowe określenie wielkości niespójności danych. W tym przypadku należy wykorzystać tabelę (4.1) i wyznaczyć współczynnik nazywany *Consistency Ratio* (CR) według wzoru:

$$CR(A) = \frac{CI(A)}{RI_n}$$

Tabela 4.1. Wartości RI_n

n	3	4	5	6	7
RI_n	0.5247	0.8816	1.1086	1.2479	1.3417

W przypadku naszej macierzy M wynosi on: $\frac{0.159759}{0.5247} = 0.3044768$. W tej metodzie również przyjmuje się, że warunkiem spójności jest spełnienie nierówności: $CR \leq 0.1$, więc klasyfikujemy macierz M jako niespójną.

4.3. Metoda odległościowa - Koczkodaj

Jedną z głównych wad współczynnika Saaty'ego jest fakt, że wartości własne są wielkościami charakteryzującymi całą macierz, nie pozwalają więc określić, które elementy powodują wystąpienie niespójności. Rozwiązaniem tego problemu jest wprowadzona przez Koczkodaja [10] metoda odległościowa, którą krótko przedstawię. Nieco bardziej rozbudowany opis, napisany przyjaznym językiem, można znaleźć w [11].
Zacznijmy od rozważenia macierzy o wymiarach 3×3:

$$A = \begin{pmatrix} 1 & a & b \\ \frac{1}{a} & 1 & c \\ \frac{1}{b} & \frac{1}{c} & 1 \end{pmatrix}$$

Odwołując się do 4.1 możemy wnioskować, że $b = ac$. Nasza macierz będzie spójna, jeśli spełniony zostanie ten warunek.

W celu zmierzenia ewentualnej niespójności, ponownie wykorzystując 4.1, możemy stworzyć trzy wektory, w których jedna wartość zostanie wyliczona jako kombinacja dwóch pozostałych. Otrzymujemy więc wektory: $(\frac{b}{c}, b, c)$, (a, ac, c) i $(a, b, \frac{b}{a})$. Następnie sprawdzamy odległość każdego z tych wektorów od danego w przykładzie wektora (abc). Wybieramy ten, którego wartość odległości jest najniższa. Uzyskany rezultat to współczynnik niespójności. Zapis formalny przedstawionego algorytmu wygląda następująco:

$$CM(a,b,c) = min\{\frac{1}{a}|a - \frac{b}{c}|, \frac{1}{b}|b - ac|, \frac{1}{c}|c - \frac{b}{a}|\}$$

Teraz możemy przejść do macierzy o większych wymiarach. Okazuje się, że wystarczy wyszukać wszystkie trójki liczb, które powinny być od siebie zależne. Trójki te zostały nazwane *triadami*. Aby wyznaczyć współczynnik niespójności macierzy wystarczy wybrać triad, dla którego wyliczona wartość jest największa:

$$CM(A) = max\{\{|1 - \frac{b}{ac}|, |1 - \frac{ac}{b}|\} \qquad \forall_{triady \quad (a,b,c) \quad macierzy \quad A}$$

Przyjmuje się, że macierz jest spójna, jeżeli $CM(A) \leq \frac{1}{3}$.

Warto zauważyć, że metoda odległościowa pozwana nie tylko zbadać niespójność, ale także wskazuje miejsce, które ma na nią największy wpływ. Dlatego można poprawić wprowadzone do macierzy PC wartości w jednym, konkretnym miejscu i przez to zmniejszyć współczynnik niespójności.

4.4. Biblioteka PairwiseComparisons - niespójność

Funkcje z biblioteki PairwiseComparisons pomocne w powyższych obliczeniach:

Współczynnik Saaty'ego

Nazwa

saatyIdx

Opis

Oblicza współczynnik niespójności macierzy w podstawowej wersji zaproponowanej przez Saaty'ego.

Argumenty

matrix – PC matrix

Zwracana wartość

Wartość współczynnika Saaty'ego.

Dodatkowe informacje

W bibliotece dostępna jest również symboliczna wersja tej funkcji.*saatyIdxSym*

Współcznnik niespójności Koczkodaja dla triady

Nazwa

koczkodajTriadIdx

Opis

Oblicza współczynnik niespójności dla triady metodą Koczkodaja

Argumenty

triad – wektor trzech liczb

Zwracana wartość

Współczynnik Koczkodaja

Najbardziej niespójny triad macierzy

Nazwa

koczkodajTheWorstTriad

Opis

Znajduje triad, którego wartość współczynnika niespójności Koczkodaja jest największa.

Argumenty

matrix – PC matrix

Zwracana wartość

Triad o największym współczynniku niespójności.

Najbardziej niespójne triady macierzy

Nazwa

koczkodajTheWorstTriads

Opis

Znajduje triady, których wartości współczynnika niespójności Koczkodaja są największe.

Argumenty

matrix – PC matrix

n – ilość poczukiwanych triad

Zwracana wartość

Triady o największym współczynniku niespójności.

Współczynnik niespójności Koczkodaja

Nazwa

koczkodajIdx

Opis

Oblicza współczynnik niespójności macierzy w podstawowej wersji zaproponowanej przez Koczkodaja.

Argumenty

matrix – PC matrix

Zwracana wartość

Wartość współczynnika Koczkodaja.

Spójny triad

Nazwa

koczkodajConsistentTriad

Opis

Na podstawie przekazanej trójki liczb, znajduje triad , którego wartość współczynnika niespójności jest najmniejsza.

Argumenty

triad – wektor trzech liczb

Zwracana wartość

Triad o najmniejszym współczynniku niespójności.

Poprawiona macierz

Nazwa

koczkodajImprovedMatrixStep

Opis

Znajduje miejsce (triad), w którym macierz jest najbardziej niespójna, a następnie poprawia wartości w tych miejsach, w taki sposób, aby współczynnik niespójności zmniejszył nię

Argumenty

matrix – PC matrix

Zwracana wartość

Macierz o mniejszym współczynniku niespójności.

5. Pozostałe metody

W ciągu wielu lat rozwijania metody porównań parami i badań prowadzonych w tej dziedzinie, powstało wiele funkcji, które nie są bezpośrednio częścią PC, przyczyniają się jednak do weryfikacji prawidłowości działania metody, upraszczają pracę z macierzami, również tymi niekompletnymi, czy pozwalają sprawdzić otrzymane wyniki. W tym rozdziale przedstawię kilka z takich metod.

5.1. Łączenie rankingów

Kiedy chcemy rozwiązać jakiś określony problem wykorzystując metodę porównań parami, możemy sami dokonać porównań lub poprosić o nie ekspertów w danej dziedzinie. To daje nam nadzieję, że wyniki będą obiektywne i odzwierciedlające rzeczywistość. Po otrzymaniu od nich macierzy PC, chcemy połączyć te tabele i utworzyć z nich jedną, która będzie odzwierciedlać preferencje wszystkich ekspertów. Drugą alternatywą jest obliczenie wektora wag dla każdej z otrzymanych macierzy, a następnie połączenie ich w jeden, sumaryczny wektor.

Aby tego dokonać możemy posłużyć się metodami *Aggregating individual judgments and priorities (AIJ i AIP)*, które wprowadzili Forman i Peniwati [12]. Proponują oni użycie funkcji, które wyliczają średnie arytmetyczne i geometryczne, zarówno dla macierzy, jak i dla wektorów.

Przyklad 5.1.1.

$$M_1 = \begin{pmatrix} 1 & 2 & 4 \\ \frac{1}{2} & 1 & 2 \\ \frac{1}{4} & \frac{1}{2} & 1 \end{pmatrix} \qquad M_2 = \begin{pmatrix} 1 & \frac{1}{3} & 4 \\ 3 & 1 & 9 \\ \frac{1}{4} & \frac{1}{9} & 1 \end{pmatrix}$$

Obliczamy macierz PC, która odzwierciedla obie macierze:

– *z wykorzystaniem średniej arytmetycznej:*

$$M_{art} = \begin{pmatrix} 1 & \frac{7}{6} & 4 \\ 1\frac{3}{4} & 1 & 5\frac{1}{2} \\ \frac{1}{4} & \frac{11}{36} & 1 \end{pmatrix}$$

– *z wykorzystaniem średniej geometrycznej:*

$$M_{geom} = \begin{pmatrix} 1 & 0.8164966 & 4 \\ 1.224745 & 1 & 4.242641 \\ 0.25 & 0.2357023 & 1 \end{pmatrix}$$

5.2. Wektor wag a macierz PC

Przyjrzyjmy się jeszcze raz przedstawionemu już wcześniej warunkowi spójności macierzy: $m_{ik} = m_{ij}m_{jk}$. Wyprowadzenie tego wzoru jest proste i opiera się na spostrzeżeniu, że stosunek dwóch elementów wektora wag powinien być równy odpowiadającej im wartości w macierzy PC. Przykładowo, jeśli alternatywa druga ma wartość 0.4, a alternatywa trzecia 0.2, to spodziewamy się, że wartość porównania drugiej i trzeciej alternatywy wynosi $\frac{0.4}{0.2} = 2$. Szczegółową wersję spójności macierzy można więc przedstawić następująco:

$$a_{ij}a_{jk} = \frac{w_i}{w_j}\frac{w_j}{w_k} = \frac{w_i}{w_k} = a_{ik}. \tag{5.1}$$

Właśnie na tym spostrzeżeniu bazują Bana e Costa i Vansnick [13] w swoich badaniach, wprowadzając dwa sposoby sprawdzania spójności i wykrywania miejsc, w których macierz nie jest spójna: *(first and second Condition of Order Preservation - COP)*. Pierwszy sposób (COP1) sprawdza, czy wybór lepszej alternatywy w każdej parze przekłada się na większą wartość danej alternatywy w wektorze wag. Drugi sposób (COP2) jest bardziej dokładny i wprost weryfikuje warunek 5.1 dla każdej pary alternatyw.

5.3. Odległość między wektorami

Ciekawym algorytmem jest obliczanie odległości między wektorami. Sposób ten przedstawił Maurice Kendall [14]. Opiera się on na idei sortowania bąbelkowego. W czasie sortowania tego rodzaju, kolejno porównujemy sąsiadujące ze sobą elementy i jeśli nie są we właściwej kolejności, to zamieniamy je miejscami. Czynność powtarzamy aż do momentu, gdy cała lista elementów jest posortowana. Kendall zaproponował algorytm zliczający ilość *zamian miejscami*, które należałoby wykonać w danym wektorze, by stał się identyczny do drugiego wektora.

5.4. Inne funkcje usprawniające pracę z macierzami PC

W bibliotece PairwiseComparisons znalazły się także inne funkcje, które bezpośrednio nie dotyczą metody porównań parami, ułatwiają jednak prace z macierzami, w szczególności z macierzami PC. Przykładem może być funkcja *recreatePCMatrix*, która przyjmuje macierz z uzupełnionymi wartościami tylko powyżej przekątnej, a pozostałe elementy uzupełnia automatycznie.

5.5. Biblioteka PairwiseComparisons - pozostałe metody

Funkcje z biblioteki PairwiseComparisons pomocne w powyższych obliczeniach:

Agregacja macierzy/wektorów (średnia arytmetyczna)

Nazwa

AIJadd

Opis

Oblicza macierz lub wektor, którego elementy są średnią arytmetyczną przekazanych macierzy lub wektorów.

Argumenty

... – lista macierzy lub wektorów tych samych wymiarów

Zwracana wartość

Średnia macierz/wektor

Agregacja macierzy/wektorów (średnia geometryczna)

Nazwa

AIJgeom

Opis

Oblicza macierz lub wektor, którego elementy są średnią geometryczną przekazanych macierzy lub wektorów.

Argumenty

... – lista macierzy lub wektorów tych samych wymiarów

Zwracana wartość

Średnia macierz/wektor

Lista COP1

Nazwa

cop1ViolationList

Opis

Wyznacza listę indeksów macierzy, które nie spełniają pierwszego warunku *Condition of Order Preservation (COP1)*

Argumenty

matrix – macierz PC

resultList – wektor wag macierzy

Zwracana wartość

Lista indeksów niespełniających $COP1$

COP1

Nazwa

cop1Check

Opis

Sprawdza czy każda para indeksów macierzy spełnia pierwszy warunek *Condition of Order Preservation (COP1)*

Argumenty

matrix – macierz PC

resultList – wektor wag macierzy

Zwracana wartość

true jeśli $COP1$ jest spełniony, w przeciwnym razie *false*

Lista COP2

Nazwa

cop2ViolationList

Opis

Wyznacza listę indeksów macierzy, które nie spełniają drugiego warunku *Condition of Order Preservation (COP2)*

Argumenty

matrix – macierz PC

resultList – wektor wag macierzy

Zwracana wartość

Lista indeksów niespełniających $COP2$

COP2

Nazwa

cop2Check

Opis

Sprawdza czy każda para indeksów macierzy spełnia drugi warunek *Condition of Order Preservation (COP2)*

Argumenty

matrix – macierz PC

resultList – wektor wag macierzy

Zwracana wartość

true jeśli $COP2$ jest spełniony, w przeciwnym razie *false*

Rozbieżność rankingu

Nazwa

errorMatrix

Opis

Oblicza rozbieżność dla każdego elementu macierzy i tworzy macierz E zawierającą elementy obliczane według wzoru $e_{ij} = m_{ij}\frac{r_i}{r_j}$. Jeśli macierz M nie zawiera elementów rozbieżnych (warunek *COP2* jest spełniony), każdy element e_{ij} wynosi 1.

Argumenty

matrix – macierz PC

resultList – wektor wag macierzy

Zwracana wartość

Macierz błędów $E = [e_{ij}]$

Lokalna rozbieżność rankingu

Nazwa

localDiscrepancyMatrix

Opis

Oblicza wielkość rozbieżności (patrz *errorMatrix*) dla każdego elementu.

Argumenty

matrix – macierz PC

resultList – wektor wag macierzy

Zwracana wartość

Macierz z lokalnymi rozbieżnościami

Globalna rozbieżność rankingu

Nazwa

globalDiscrepancy

Opis

Znajduje największą lokalną rozbieżność (patrz *localDiscrepancyMatrix*).

Argumenty

matrix – macierz PC

resultList – wektor wag macierzy

Zwracana wartość

Maksymalna wartość lokalnej rozbieżności

Odległość między wektorami

Nazwa

kendallTauDistance

Opis

Oblicza odległość Kendall Tau (sortowanie bąbelkowe) pomiędzy dwoma wektorami.

Argumenty

list1 - pierwszy wektor do porównania
list2 - drugi wektor do porównania

Zwracana wartość

Liczba *zamian miejscami*, które należy wkonać, aby kolejność elementów w wektorach była identyczna

Znormalizowana odległość między wektorami

Nazwa

normalizedKendallTauDistance

Opis

Oblicza odległość Kendall Tau (sortowanie bąbelkowe) pomiędzy dwoma wektorami (patrz *kendallTau-Distance* i dzieli je przez liczbę wszystkich możliwych *zamian miejscami*.

Argumenty

list1 - pierwszy wektor do porównania
list2 - drugi wektor do porównania

Zwracana wartość

Stosunek liczba *zamian miejscami*, które należy wkonać, aby kolejność elementów w wektorach była identyczna do liczby wszystkich możliwych *zamian miejscami*

Usuń wiersze

Nazwa

deleteRows

Opis

Usuwa wybrane wiersze z macierzy

Argumenty

matrix – macierz PC
listOfRows – wektor indeksów wierszy, które należy usunąć

Zwracana wartość

Macierz po usunięciu wskazanych wierszy

Usuń kolumny

Nazwa

deleteColumns

Opis

Usuwa wybrane kolumny z macierzy

Argumenty

matrix – macierz PC

listOfColumns – wektor indeksów kolumn, które należy usunąć

Zwracana wartość

Macierz po usunięciu wskazanych kolumn

Usuń wiersze i kolumny

Nazwa

deleteRowsAndColumns

Opis

Usuwa wybrane wiersze i kolumny z macierzy

Argumenty

matrix – macierz PC

listOfRowsAndColumns – wektor indeksów wierszy i kolumn, które należy usunąć

Zwracana wartość

Macierz po usunięciu wskazanych wierszy i kolumn

Zwróć element macierzy

Nazwa

getMatrixEntry

Opis

Zwraca $[r, c]$ element z macierzy.

Argumenty

matrix – macierz PC

r – numer rzędu

c – numer kolumny

Zwracana wartość

Wskazany element macierzy

Utwórz kompletną macierz PC

Nazwa

recreatePCMatrix

Opis

Na podstawie macierzy z uzupełnionymi wartościami nad przekątną, tworzy kompletną macierz PC

Argumenty

matrix – macierz PC

Zwracana wartość

Kompletna macierz PC

Puste elementy macierzy

Nazwa

harkerMatrixPlaceHolderCount

Opis

Sprawdza ile elementów w rzędzie ma wartość 0.

Argumenty

matrix – macierz PC

row – numer rzędu, którego elementy są sprawdzane

Zwracana wartość

Ilość pustych elementów w wierszu

Napraw macierz

Nazwa

harkerMatrix

Opis

Tworzy macierz gotową do użycia w metodach porównań parami. Niewłaściwe elementy zastępowane są wartością 0, a na przekątnej ustawiana jest wartość 1.

Argumenty

matrix – macierz z możliwymi błędnymi wartościami

Zwracana wartość

Macierz PC gotowa do użycia w metodach porównań parami.

Utwórz spójną macierz

Nazwa

consistentMatrixFromRank

Opis

Na podstawie rankingu wag W tworzy spójną macierz PC, której elemnty to $m_{ij} = \frac{w_i}{w_j}$.

Argumenty

rankList – wektor wag macierzy

Zwracana wartość

Spójna macierz PC

Sortuj ranking

Nazwa

rankOrder

Opis

Sortuje malejąco wartości wektora wag od najwyższej do najniższej

Argumenty

rankList – wektor wag macierzy

Zwracana wartość

Posortowany wektor wag macierzy

6. Podsumowanie i wnioski

Cel pracy, którym było stworzenie biblioteki PairwiseComparisons, został osiągnięty. Powstała biblioteka, zgodnie z założeniami, pokrywa funkcjonalność pakietu [6] i służy do wykonywania obliczeń matematycznych, na których opiera się metoda porównań parami. Biblioteka pozwala obliczać gotowe rankingi poprzez wywołanie tylko jednej funkcji, jak również udostępnia wiele metod, które umożliwiają stopniowe budowanie rankingów.

Pakiet napisany został w języku R, który przeznaczony jest do obliczeń matematycznych i wykorzystywany głównie w tym celu. Jego składnia i sposób definiowania własnych funkcji jest intuicyjny, a sposób działania szybki. Powstała również biblioteka napisana w języku Java umożliwiająca korzystanie z funkcji z poziomu maszyny wirtualnej Javy. Wykorzystuje bibliotekę *RCaller* [15], która pozwala na wywoływanie kodu źródłowego napisanego w języku R - przeznaczonego przecież właśnie do obliczeń. Rozwiązanie to pozwoliło w prosty sposób połączyć oba języki. Dzięki niemu nie musiałem wprost implementować szczegółów metody porównań parami w języku Java. Wykorzystanie dobrodziejstw R w pakiecie Javy okazało się dobrym wyborem i uświadomiło mi, że w ramach pisania kodu źródłowego w jednym języku, warto czasem pewne specyficzne fragmenty *odesłać* do innego, aby ułatwić implementację i usprawnić pracę całego rozwiązania.

Biblioteka może posłużyć osobom, które znają już metodę porównań parami i pracują z nią na co dzień. Powinna przyspieszyć ich działania i pozwolić skupić się na rozwijaniu metody, a nie wyszukiwaniu sposobów na zaimplementowanie podstawowych funkcjonalności w języku Java. Pakiet może zostać również wykorzystany do wprowadzenia nowych osób w zagadnienie metody porównań parami poprzez zaprezentowanie prostych przykładów jej działania. Kiedy opowiada się o PC, jednym z elementów powodującym zniechęcenie są obliczenia matematyczne, których implementacja może wydawać się skomplikowane. Powstała biblioteka pozwoli odwrócić od niej uwagę i pokazać, że można korzystać z metody w prosty sposób. Mam nadzieję, że wpłynie to pozytywnie na odbiór metody, pozwoli zwiększyć jej popularność i zainteresować nią nowe osoby.

Metoda porównań parami ciągle się rozwija i z pewnością w przyszłości będzie poszerzana o nowe funkcjonalności, a także wykorzystywana w coraz to nowych dziedzinach. Nowe rozwiązania mogą zostać wprowadzane do biblioteki, której kolejne wersje będą odpowiadać aktualnemu stopniowi rozwoju metody. Czekam więc na kolejne odkrycia w tej dziedzinie, aby zaimplementować je w językach R i Java, a następnie dostarczyć badaczom w celu usprawnienia ich pracy.

Bibliografia

[1] G. T. Fechner. *Elements of psychophysics*. T. 1. New York: Holt, Rinehart i Winston, 1966.

[2] L. L. Thurstone. „A law of comparative judgment, reprint of an original work published in 1927". W: *Psychological Review* 101 (1994), s. 266–270.

[3] T. L. Saaty. „Relative Measurement and Its Generalization in Decision Making. Why Pairwise Comparisons are Central in Mathematics for the Measurement of Intangible Factors. The Analytic Hierarchy/Network Process". W: *Estadística e Investigación Operativa / Statistics and Operations Research (RACSAM)* 102 (November 2008), s. 251–318.

[4] K. Kułakowski. „Heuristic Rating Estimation Approach to The Pairwise Comparisons Method". W: *Fundamenta Informaticae* 133.4 (2014), s. 367–386.

[5] K. Kułakowski. „Notes on the existence of a solution in the pairwise comparisons method using the heuristic rating estimation approach". W: *Annals of Mathematics and Artificial Intelligence* 77.1 (2016), s. 105–121.

[6] K. Kułakowski. *http://home.agh.edu.pl/~kkulak/doku.php?id=user:konrad:researchlibs:pcpackage*.

[7] T. L. Saaty. „A scaling method for priorities in hierarchical structures". W: *Journal of Mathematical Psychology* 15.3 (1977), s. 234–281.

[8] M. Burnelli. *Introduction to the Analytic Hierarchy Process*. SpringerBriefs in Operations Research, 2015.

[9] K. Kułakowski. „A heuristic rating estimation algorithm for the pairwise comparisons method". W: *Central European Journal of Operations Research* 23.1 (2015), s. 187–203.

[10] W. W. Koczkodaj. „A new definition of consistency for pairwise comparisons". W: *Mathematical and Computer Modelling* 18.7 (1993), s. 79–84.

[11] K. Wójcik. *Portal ankiet porównawczych*. AGH University of Science i Technology.

[12] E. H. Forman i K. Peniwati. „Aggregating Individual Judgments and Priorities with the AHP". W: *European Journal of Operational Research* 108.1 (1998), s. 165–169.

[13] C. A. Bana e Costa i Jean-Claude Vansnick. „A critical analysis of the eigenvalue methodused to derive priorities in AHP". W: *European Journal of Operational Research* 187.3 (2008), s. 1422–1428.

[14] Kendall tau distance. *https://en.wikipedia.org/wiki/Kendall_tau_distance*.

[15] M. H. Satman. „RCaller: A Software Library for Calling R from Java”. W: *British Journal of Mathematics Computer Science* 4.15 (2014), s. 2188–2196.