***Metody Sztucznej Inteligencji w Mechanice Płynów***

***Redukcja wymiarowości, linearyzacja***

Nazywam się Kornel mrozowski i będę mówił o metodach sztucznej inteligencji które znajdują swoje zastosowanie w mechanice płynów.

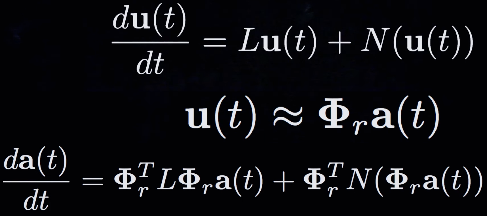
Na ostatniej prezentacji

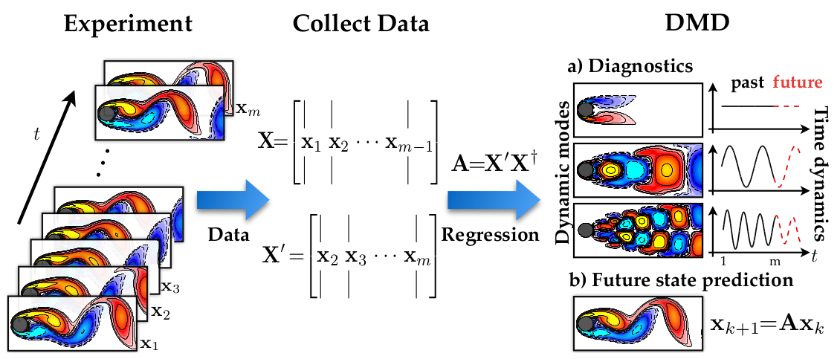
Na ostatniej prezentacji pokazałem jak można zastosować sieć neuronową typu encoder decoder do pobrania wartości i wektorów własnych z pewnej macierzy. Każdy wektor własny otrzymany taką metodą po przemnożeniu przez odpowiadającą mu wartość własną jest interpretowany np. jako składowa wartości ciśnienia lub prędkości.

Dzisiaj chciałem omówić 2 metody sztucznej inteligencji związane z tworzeniem modeli bazujących na danych historycznych pozwalających na predykcję wielkoskalowych symulacji mechaniki płynów, np. związanych z prognozowaniem pogody na oceanach.

Dzisiaj chciałbym skupić na metodzie tworzenia modeli, które mogą się przebudowywać w zależności od wybranego parametru (tylko jednego dla prostoty, prędkości lotu, średnicy cylindra, czegokolwiek). Żeby było prościej nie będzie równań N-S, zastąpi go prosty wzór: pochodna X po czasie równa się pewnej nieznanej funkcji prawych stron zależnej od X-a, czasu i tego jednego parametru mi. X jest dużym wektorem zmiennych stanu np. 1mln-elementowym. Możemy też mierzyć kilka współrzędnych X-a np. 10 elementów. Przypuśćmy, że dane z symulacji, którymi dysponujemy dotyczą Boeinga 787 i chcemy znać rozkład ciśnienia wokół np. jego podwozia przy lądowaniu, ponieważ z tego rozkładu wynikają opory, bardzo istotne przy lądowaniu samolotu. Dysponujemy kilkoma rurkami Prandtla, rozmieszczonymi na skrzydłach, dziobie, na lotkach. Rurki nie są idealne, mają pewien nieznany błąd.

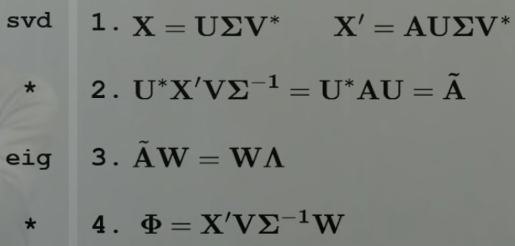
[slajd]

To równanie opisuje nasz system dynamiczny. Jest podzielony na część liniową i nieliniową. Przypuśćmy, że jest on parametryzowany parametrem . I kiedy on się zmienia model otrzymany za pomocą sieci, którą omówiłem na poprzedniej prezentacji nie działa. Dla przypomnienia był to model, który przybliżał aktualną chwilę symulacji w dowolnym czasie korzystając z wektorów własnych pochodzących z nagrania eksperymentu kamerą w metodzie PIV. To, że on nie działa jest typowe dla zadań uczenia maszynowego, szkolisz model na jednych danych, a kiedy na wejściu dostaje dane, których nigdy nie widział to model jest niedokładny, albo wcale nieprzydatny. Dlatego zadanie ekstrapolacji jest trudne, ale do rozwiązania. Wystarczy mutować model w zależności, w jakich okolicznościach przychodzi mu działać.

[slajd]

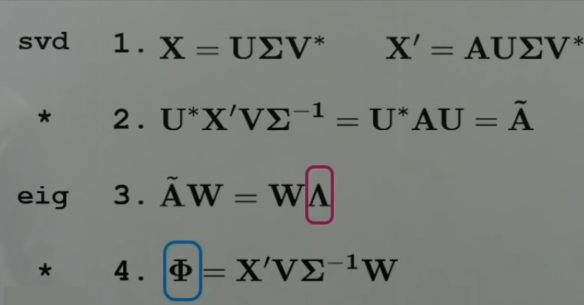
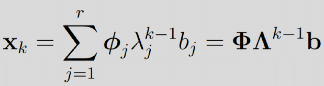
Powiedzmy że przeprowadziliśmy symulacje dla ustalonego parametru w 1000 chwilach czasowych. Teraz każdą chwilę czasową tej symulacji zamieniamy kolumnę macierzy X, a wszystkie otrzymane wartości temperatury, ciśnienia, prędkości, itd. w danej chwili czasowej są elementami tej kolumny, powiedzmy, że to jest 1mln zmiennych stanu. Szukamy takiej macierzy A, która przeniesie nas w przyszłość…, dosłownie, czyli aktualny wektor stanu X po przemnożeniu przez to macierz daje nam wektor stanu z chwili następnej X’. Jednak macierz A, która w naszym problemie byłaby 1mln na 1mln i nie zmieściłaby się na dysku rodzi potrzebę zmniejszenia wymiarowości danych. W tym celu zamiast A korzysta się z jej sfaktoryzowanej postaci otrzymanej metodą Dynamic Mode Decomposition. Czyli macierzy A~, której rząd nie jest równy liczbie zmiennych przechowywanych w kolumnie X, a ilości wykonanych kroków czasowych, zatem wynosi 1000.

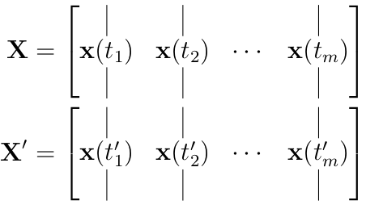
[slajd]

W pierwszym kroku algorytmu DMD wykonujemy rozkład według wartości osobliwych macierzy stanowiącej zgromadzone dane historyczne, macierzy X. Macierz U zawiera wektory własne macierzy X. Wektory te mają przyjemną właściwość że ich kombinacja liniowa kilku pierwszych z nich wystarcza, żeby przybliżyć macierz X. Nawet jeśli miałaby 1000 kolumn i milion rzędów to można wziąć 10 wektorów własnych i zrekonstruować ją.

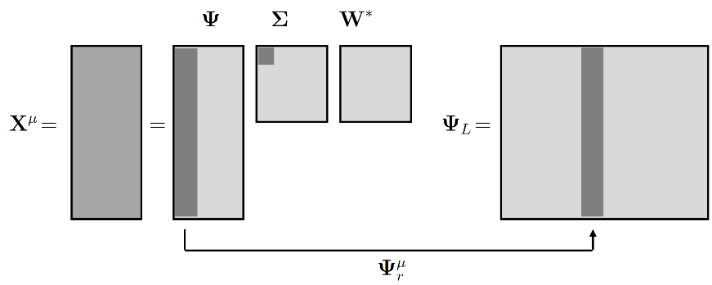
Teraz podstawiając to do wzoru możemy obliczyć macierz, A. Nie robimy tego, żeby nie zapchać dysku twardego. Zamiast tego rzutujemy macierz A na wektory własne X i dostajemy mniejszą macierz A~. Dzięki temu zamiast macierzy 1mln na 1mln posługujemy się macierzą 1tyś na 1tyś. Macierz A~ to najlepszy liniowy system, który spełnia nieliniowość macierzy danych X, czyli mówi jak jej wektory własne zmieniają się z czasem.

[slajd]

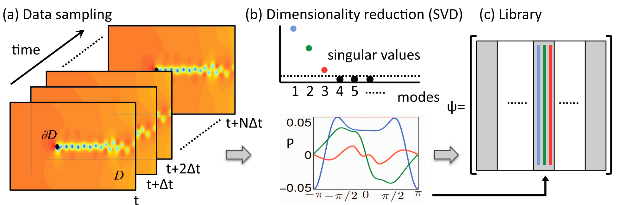
W trzecim kroku otrzymujemy wektory i wartości własne macierzy A~, która ma inne wektory, ale te same wartości własne co macierz A. Ale nam najbardziej zależy na wektorach własnych macierzy A, ponieważ one są tymi dominującymi strukturami, z których składa się przepływ. W taki sposób możemy przybliżyć każdy przyszły stan naszego symulowanego systemu. To podejście sprawdza się dobrze w systemach, które mają charakter powtarzalny, zaś w systemach zmieniających się z czasem, czyli w niestacjonarnych systemach metoda ta nie działa i trzeba bardziej generalizującej metody.

[slajd]

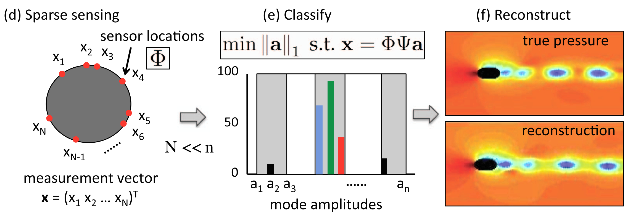
W wyniku wykonanego algorytmu dostaliśmy macierze o następującym kształcie. [slajd] Bardziej generalne podejście będzie bazowało na wybraniu tylko kilku wektorów własnych.

[slajd] oraz umieszczeniu ich w bibliotece z wektorami własnymi.

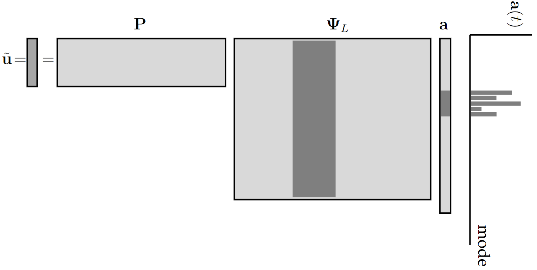
Jak wspomniałem nasze równanie które opisuje rozważany system możemy sparametryzować poprzez stałą . Dla każdej tej stałej wykonujemy przedstawiony wcześniej algorytm DMD i wartości własne powstałe w wyniku wykonywania go magazynujemy w bibliotece z wektorami własnymi. W ten sposób można uniewrażliwić model na przewidziane z góry zmiany, które mogą zajść w naszym fizycznym systemie opisywanym przez układ równań różniczkowych, ponieważ możemy teraz wykryć którym systemie fizycznym znajduje się model a raczej nowy polimorficzny model tego dokona i wybierze z biblioteki te wektory własne, które zostały otrzymane dla tego wariantu systemu. Tylko teraz jak zdecydować, których wektorów własnych używać? To będzie zadanie klasyfikacji.

[slajd]

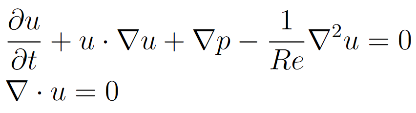
Żeby spojrzeć z perspektywy i zobaczyć, gdzie się właśnie znajdujemy to… (a) przetworzyliśmy zebrane dane, (b) znaliśmy wartości i wektory własne (c) i zbudowaliśmy z nich bibliotekę.

[slajd]

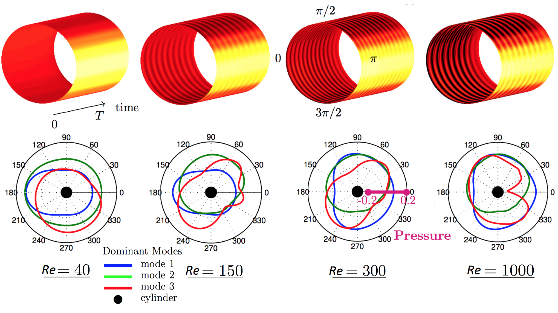
Teraz rozmieścimy czujniki badające zmienne stanu w wybranych punktach. Umożliwią nam one klasyfikację z jakich wektor własnych powinniśmy budować model. Po wybraniu grupy wektorów własnych i zbudowaniu modelu możemy dokonać rekonstrukcji przepływu w każdym punkcie na podstawie danych pomiarowych z tylko kilku wybranych punktów.

[slajd]

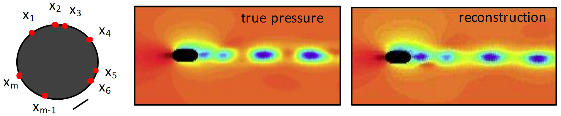
Znamy wektor danych pomiarowych, miejsca, w których mierzymy, mamy bibliotekę wektorów własnych, to możemy policzyć wektor „a”. Niezerowe elementy tego wektora wskazują, z których wektorów własnych należy zbudować model oraz stanowią przepis w jakich proporcjach, który wektor należy wziąć do modelu. Chciałbym teraz pokazać przykład działania tej metody dla przepływu wokół walca, który opisuje taki wzór:



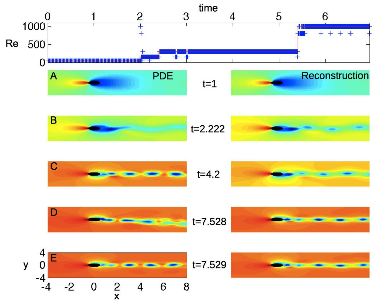
[slajd]

Wynikiem rekonstrukcji są 4 walce, wzdłuż których zaprezentowano zmiany ciśnienia w czasie. Wokół nich wystąpiły 4 przepływy z 4 różnymi liczbami Reynoldsa czyli z 4 różnymi prędkościami opływającej wody. Liczba Reynoldsa to właśnie jest ta stała, wcześniej oznaczana jako , dla której budowana była biblioteka. Przedstawione wcześniej kroki znalazły następujące wektory własne. Każdy z nich jest rozkładem ciśnienia na ściance walca.

[slajd]

Okazuje się że taki przebudowywujący się model, który znalazł oczekiwane rozkłady ciśnień pozwala zrekonstruować przepływ wokół walca z bardzo dobrą dokładnością, ok 99,9%. 

[slajd]

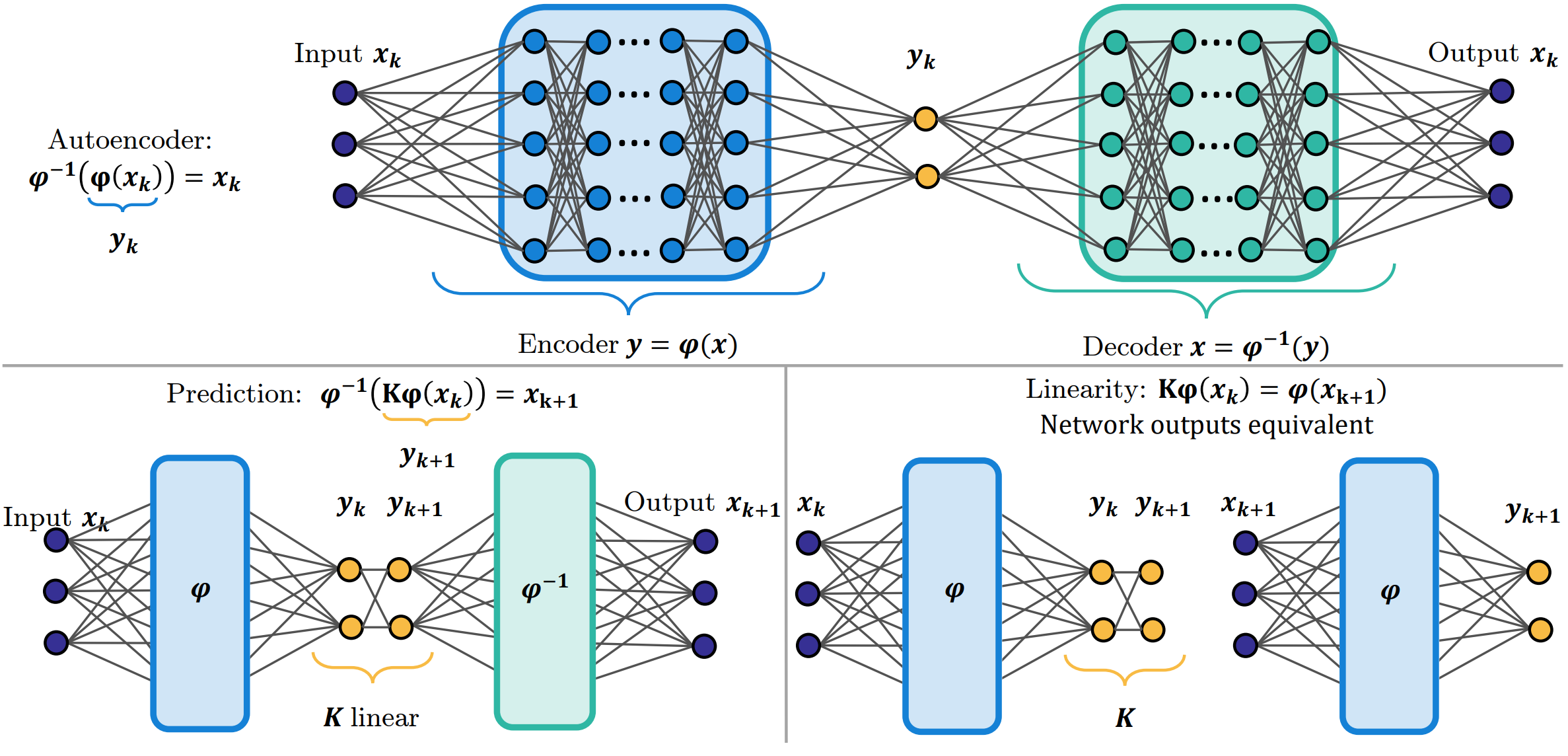
Linia ciągła na górnym wykresie pokazuje rzeczywistą liczbę Reynoldsa użytą w pełnej symulacji. Krzyrzyki to wyniki klasyfikacji wektorów własnych, a tym samym liczby Reynoldsa. Panele A-D po lewej stronie przedstawiają symulowane pole ciśnienia, a po prawej jego rekonstrukcję w czterech różnych miejscach w czasie. Dla wyższych liczb Reynoldsa klasyfikacja staje się trudniejsza.

[slajd]

Także to było zastosowanie metody Dynamic Mode Decomposition w problemie Compressive Sensing Identyfication. Teraz jeśli starczy czasu to chciałbym szybko pokazać metodę pozwalającą całkowicie zastąpić rozwiązywanie układów cząstkowych równań różniczkowego w celu poznania przyszłych stanów systemu dynamicznego. Jest to metoda prowadzenia prognoz systemów, które są za duże do zasymulowania nawet na superkomputerze. Bazuje tylko na historycznych danych pomiarowych.

Chcielibyśmy znów znaleźć taką macierz K, która po przemożeniu przez obecny stan zabierze nas do przyszłego stanu. Przeszkodą na tej drodze jest nieliniowość równań różniczkowych. Za pomocą sieci neuronowej typu auto-enkoder można znaleźć taką transformację, która zmieniania nieliniowy problem liniowy.

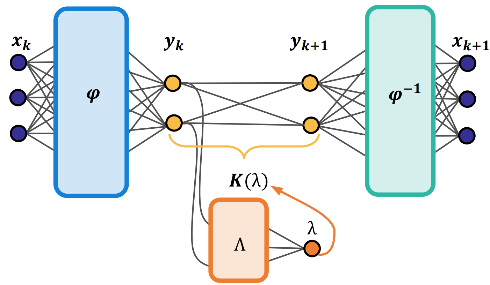
Jest taki operator Koopmana, wymyślony w 1931 roku, jest macierzą, która umożliwia przejście ze stanu aktualnego do przyszłego, ale tylko dla zmiennych stanu przeniesionych z układu, w którym ich zachowanie jest nieliniowe do takiego układu współrzędnych, w którym jest ono liniowe. Innymi słowy, funkcje własne, czy też wektory własne tego operatora globalnie linearyzują dynamikę. Jednak identyfikacja i reprezentacja tych funkcji własnych okazuje się być wyzwaniem obliczeniowym. W tym polu zastosowanie znalazło głębokie uczenie, które znajduje funkcje własne Koopmana na podstawie danych historycznych systemów dynamicznych.



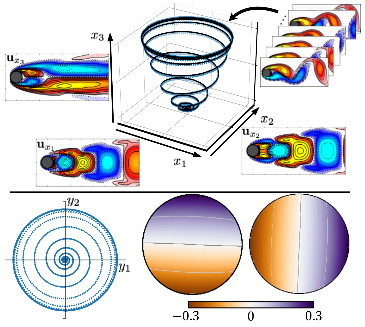
Przedstawiona sieć typu auto-encoder szuka takiego układu współrzędnych których dynamika systemu jest globalnie liniowa. Obszar zastosowań metody którą przedstawię znajduje zastosowanie w każdym fizycznym nieliniowym zjawisku, np. w optyce, w modelowaniu turbulencji, nawet w prostym wahadle matematycznym. Korzystamy z zalet generalizowania problemów przez głębokie sieci neuronowe oraz z fizycznej interpretowalności operatorów Koopmana.

Na początku, zakładamy że **.** D, gdzie **x** to wektor zmiennych stany w chwili **k,** a **F** mapuje stan poprzedni z następnym. Funkcje własne są jednocześnie nowymi współrzędnymi, których szukamy. Górny schemat przedstawia proces uczenia sieci którym otrzymujemy funkcje własne, kodujące oraz ich odwrotności, służące do dekodowania. Jako dane trenujące używamy wejście do sieci. Następnie zmienia się konstrukcję sieci na taką, która umożliwi znalezienie operatora Koopmana. Wejściem do sieci zostaje to co było, zaś jej wyjściem jest już wektor stanu w chwili następnej. Ten 2 architektury sieci stosujemy do predykcji przyszłych stanów analizowanego dynamicznego nieliniowego systemu fizycznego. I to się sprawdza i się stosuje. Tylko, że w takiej wersji okno na przyszłość jest dosyć wąskie. Model ten nie bierze pod uwagę zmieniających się częstotliwości związanych np. z dyssypacją energii mechanicznej, czyli tarciem cząsteczek o siebie, co zamienia tą energię w ciepło. Dzieje się to analogicznie do wahadła, którego częstotliwość wahnięć spada przy wzroście amplitudy czyli w zasadzie energii mechanicznej.

[slajd]

Z tego powodu stosuje się czwartą sieć, która uczy się dobierać wartości własne operatora Koopmana. W ten sposób można uwzględnić ciągłość w zanikaniu energii układu. Innymi słowy, pozwalamy, aby wartości własne macierzy Koopmana zmieniały się λ = Λ(y). Wartości własne λ± = µ ± iω są następnie wykorzystywane do parametryzacji przekątnej operatora Koopmana K(µ, ω).

[slajd]

Na załączonej grafice przedstawiono poznane funkcje własne Koopmana dla modelu pola przepływu o liczbie Reynoldsa 100.

U góry (góra) Rekonstrukcja trajektorii z liniowego modelu Koopmana z dwoma stanami; tryby dla każdej ze zmiennych przestrzeni stanów x są pokazane wzdłuż osi współrzędnych. (na dole) Rekonstrukcja Koopmana we współrzędnych funkcji własnej y, wraz z funkcjami własnymi y = ϕ(x).

Wektory i wartości własne

To co łączy mechanikę płynów i uczenie maszynowe to to, że w nawet najbardziej skomplikowanych przepływach istnieją wzorce. Dla przykładu jak bym miał jedno 1MP zdjęcie twarzy człowieka to te miliony informacji nie jest koniecznie potrzebne i nie trzeba budować sieci splotowej. Są dominujące wzorce które definiują tą twarz i tak samo są wzorce które definiują pole przepływu. W 1978 roku Sirovich napisał 2 artykuły które zmieniły świat. Pierwszy z nich to tzw. twarze własne, w którym wziął bibliotekę z 1000 zdjęć ludzkich twarzy i przeprowadził na niej rozkład według wartości osobliwych aby otrzymać twarze własne lub też wektory własne tych zdjęć.

Wektory własne i wartości własne

Tutaj widać jak przykład dla klatek z fulmu przedstawiającego łączące się dwa wiry. Następnie niżej jest dekompozycja na wektory własne. A w trzecim rzędzie odtworzenie Pierwszego rzędu, oczywiście nie jest idealnie, ponieważ została użyta tylko jedna moda.

Slajd

W tym samym roku Sirovich swoją dekompozycję twarzy użył na filmach przedstawiających przepływy. Pokazał wtedy, że jeśli weźmie się wektory własne tego filmu to można otrzymać dominujące wzorce. Po pierwsze można otrzymać średnią klatkę z całego filmu. A następnie rozkład według wartości osobliwych pozwala otrzymać kolejne wektory własne. Wzorce tego przepływu są stałe w czasie, ale ich kombinacja liniowa jest zmienna w czasie. I kiedy dodamy je do siebie dla czasu t to możemy odtworzyć oryginalny przepływ w czasie t. To jest kamień węgielny nie tylko dla przepływach dwuwymiarowych ale również dla wyższych wymiarów.

POD/PCA

Okazuje się że rozkład według wartości osobliwych można przedstawić jako sieć neuronową z jedną ukrytą warstwą z liniowymi funkcjami aktywacji. Na wejściu jest wielowymiarowe pole przepływu następnie informacja jest kompresowana w środkowej warstwie. Na końcu w dekoderze odtwarzamy przepływ. Także kiedy trenujemy sieć to nasza funkcja straty uwzględnia różnicę pomiędzy wejściem a wyjściem, a także powinna zmuszać środkową warstwę do bycia jak najwęższą. Istnieją także bardziej skomplikowane struktury z nie liniowymi funkcjami aktywacji realizujące te same zadanie tylko lepiej. Jednym z pierwszych zastosowań głębokiego autoenkodera w 2002 roku była rekonstrukcji pola prędkości w pobliżu ściany w turbulentnym przepływie kanałowym przy użyciu ciśnienia i siły ścinania ściany (Milano i Koumoutsakos 2002).

Oderwanie warstwy przyściennej w kanale rozszerzającym się

Aby przedstawić to co zostało wtedy tak naprawdę otrzymane i z czego, to najpierw rozważmy typową sytuację w rurze przy powierzchni wewnętrznej. Ten obrazek przedstawia profile prędkości dla różnych gradientów ciśnienia. Pod rysunkiem są oczywiście gradienty prędkości ale z równania Prandtla wychodzi że mają one przy ścianie te same kierunki ale przeciwne zwroty.

Profil warstwy przyściennej obliczony autoenkoderem

Wyniki treningu sieci neuronowej są przedstawione na tym wykresie. Test przeprowadzono na danych nie widzianych wcześniej przez sieć w trakcie treningu ani w trakcie dekompozycji na wektory własne. I to pokazało że model dobrze generalizuje.

Particle Image Velosimetry

Particle Image Velosimetry to technika służąca do mierzenia prędkości przepływie przy pomocy wiązki laserowej oraz kamery. Niestety takie pomiary mają często wiele szumów co często przeszkadza w obliczaniu prędkości. wtedy z pomocą przychodzi nam znów dekompozycja

Slajd

Przykładowo na podstawie bazy danych zawierającej 36 ludzkich twarzy każda z nich oświetlona na 64 różne sposoby. Można uzyskać taki efekt że kiedy mamy zdjęcie przykładowej osoby zupełnie nie z tej bazy z takim wąsem to można go dzięki autoenkoderowi rozłożyć na zdjęcie bez wąsa oraz z wąsem.

Slajd

Także stosując to samo podejście do przepływów mając taki film posiadający szumy który został uzyskany na przykład dzięki Particle Image Velosimetry możemy rozłożyć go na obraz który tych szumów nie posiada oraz na sam szum. Nie będę teraz wchodził kto jak działa ten wzór który jest pod tymi obrazkami ale w telegraficznym skrócie jest to bardzo ciężki optymalizacyjne problem, który można rozwiązać przy użyciu danych.

W końcu po wykonaniu tej optymalizacji wektory własne mają dużo mniejsze szumy i dzięki temu mogą znacznie dokładniej zrekonstruować pole przepływu niż takie zaszumione wektory.

Slajd

Tylko wspomnę, że poza tym istnieją jeszcze inne zastosowania z użyciem autoenkodera. Bardzo interesujące jest zwiększanie rozdzilem