PCA & Kernel PCA

Вячеслав Боровицкий

26 февраля 2021 г.

Структура презентации

1. О методах понижения размерности
2. Применения PCA
3. Обыкновенный PCA
   1. Постановка задачи.
   2. Алгоритм.
   3. Резюме и замечания.
4. Kernel PCA
   1. Постановка задачи.
   2. Алгоритм.
   3. Резюме и замечания.
5. Еще немного про ядра
6. Полезные ссылки

О методах понижения размерности

Снижение размерности — это преобразование данных, состоящее в уменьшении числа переменных путём получения главных переменных.

— ru.wikipedia.org/wiki/Снижение_размерности

Зачем?

- Визуализация.
- Сжатие.
- Денойзинг, выделение фитч.

Как?

- PCA (проекция на аффинное многообразие).
- Kernel PCA (нелинейное преобразование + PCA).
- Вариации Kernel PCA: Isomap, LLE, Laplacian eigenmaps.
- Нейросети-автоэнкодеры.
- Многое другое, например t-SNE, UMAP, Mapper.

Применения PCA

PCA часто используется биологами для визуализации данных.

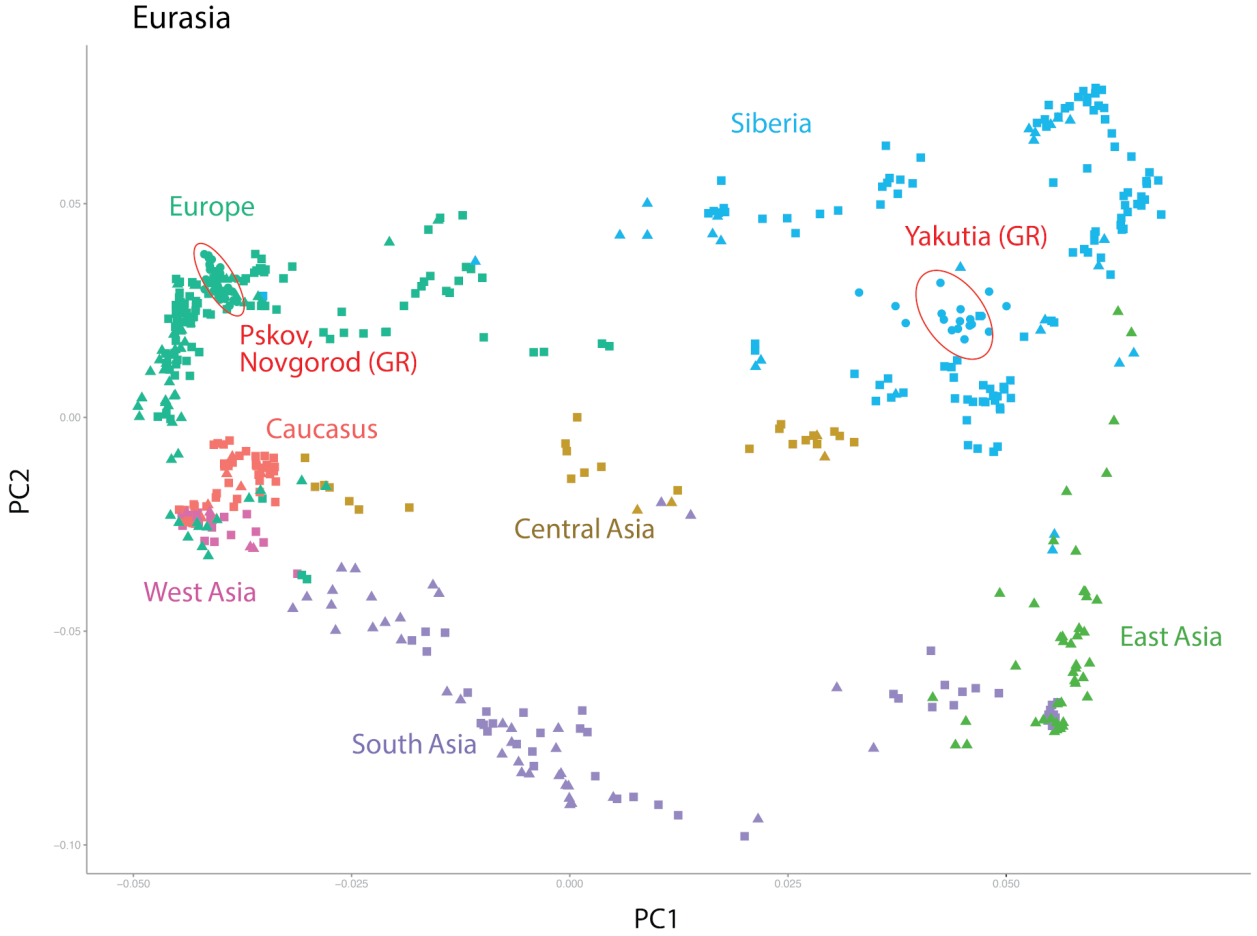
Genome-wide sequence analyses of ethnic populations across Russia

Daria V. Zhernakova [a, b], Vladimir Brukhin [a], Sergey Malov [a, c], Taras K. Oleksyk [a, d, r], Klaus Peter Koepfli [a, e], Anna Zhuk [a, f], Pavel Dobrynin [a, e], Sergei Kliver [a], Nikolay Cherkasov [a], Gaik Tamazian [a], Mikhail Rotkevich [a], Ksenia Krasheninnikova [a], Igor Evsyukov [a], Sviatoslav Sidorov [a], Anna Gorbunova [a, g], Ekaterina Chernyaeva [a], Andrey Shevchenko [a], Sofia Kolchanova [a, d], Alexei Komissarov [a], Serguei Simonov [a], Alexey Antonik [a], Anton Logachev [a], Dmitrii E. Polev [h], Olga A. Pavlova [h], Andrey S. Glotov [u], Vladimir Ulantsev [i], Ekaterina Noskova [i, j], Tatyana K. Davydova [s], Tatyana M. Sivtseva [k], Svetlana Limborska [l], Oleg Balanovsky [m, n, o], Vladimir Osakovsky [k], Alexey Novozhilov [p], Valery Puzyrev [q], Stephen J. O'Brien [a, t]

Пример: статья наших коллег из центра геномной биоинформатики им. Ф.Г. Добржанского.

Применения PCA

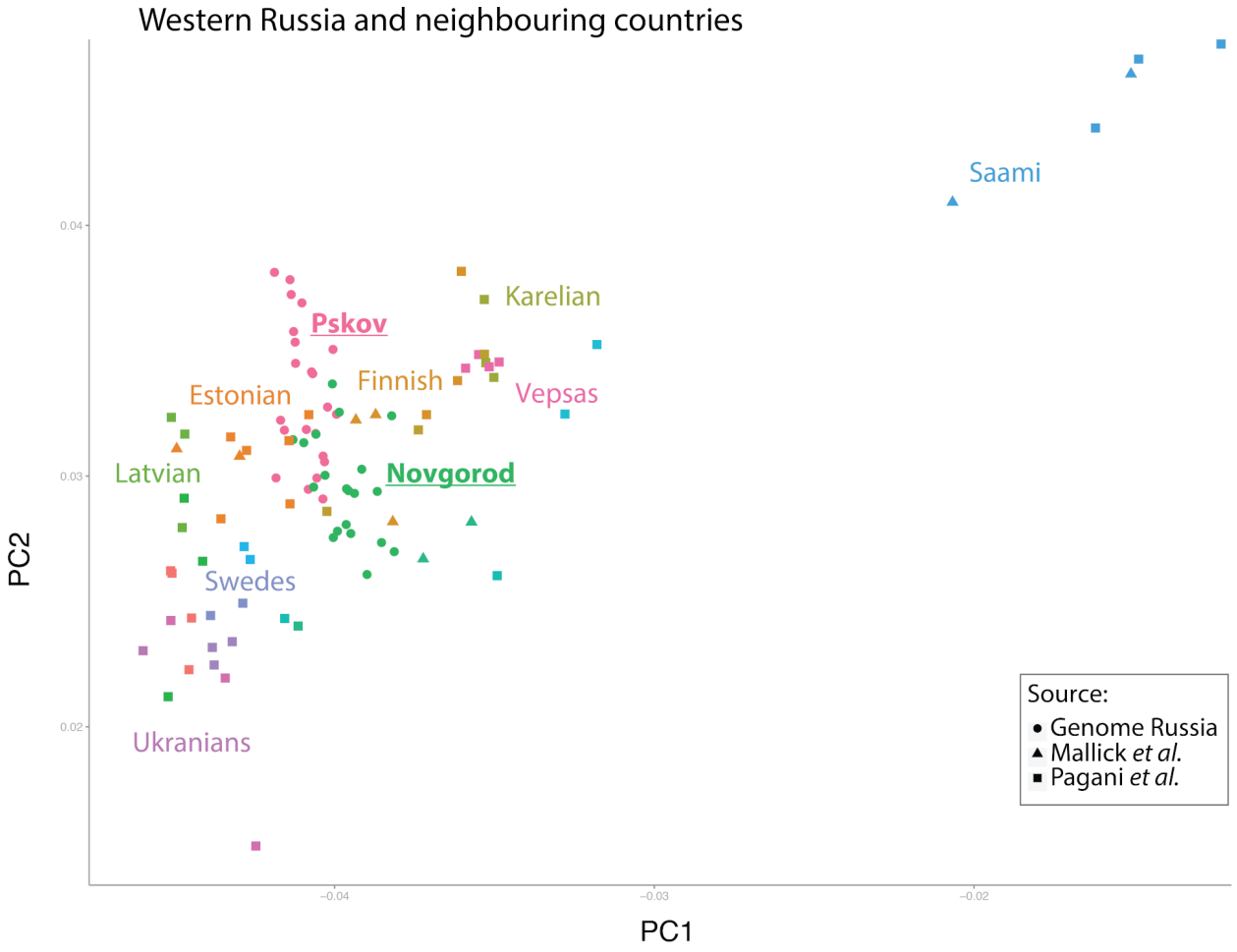
Отображаем 264 вектора размерности $\sim 10^7$ на плоскость.



PCA векторов геномных данных современных людей из разных регионов России

Применения PCA

Отображаем 310 векторов размерности $\sim 10^7$ на плоскость.



PCA векторов геномных данных современных людей с запада России и соседних стран

Применения PCA

Интерактивные примеры:

- Еще применение к геномным данным (1000 Genomes project):

  http://bwlewis.github.io/1000_genomes_examples/PCA_overview.html.

- Применение к MNIST, Iris flower data set:

  https://projector.tensorflow.org/.

Обыкновенный PCA: постановка задачи

Данные: N векторов $x_1, x_2, \ldots, x_N$ размерности D.

Цель: Найти линейное многообразие размерности P, на которое $\{x_i\}_{i=1}^N$ проектируются оптимально.

Будем считать, что x_i центрированы

$$\frac{1}{N}\sum_{n=1}^{N} x_n = 0,$$

чтобы не возиться с аффинностью и жить в “линейном” мире.

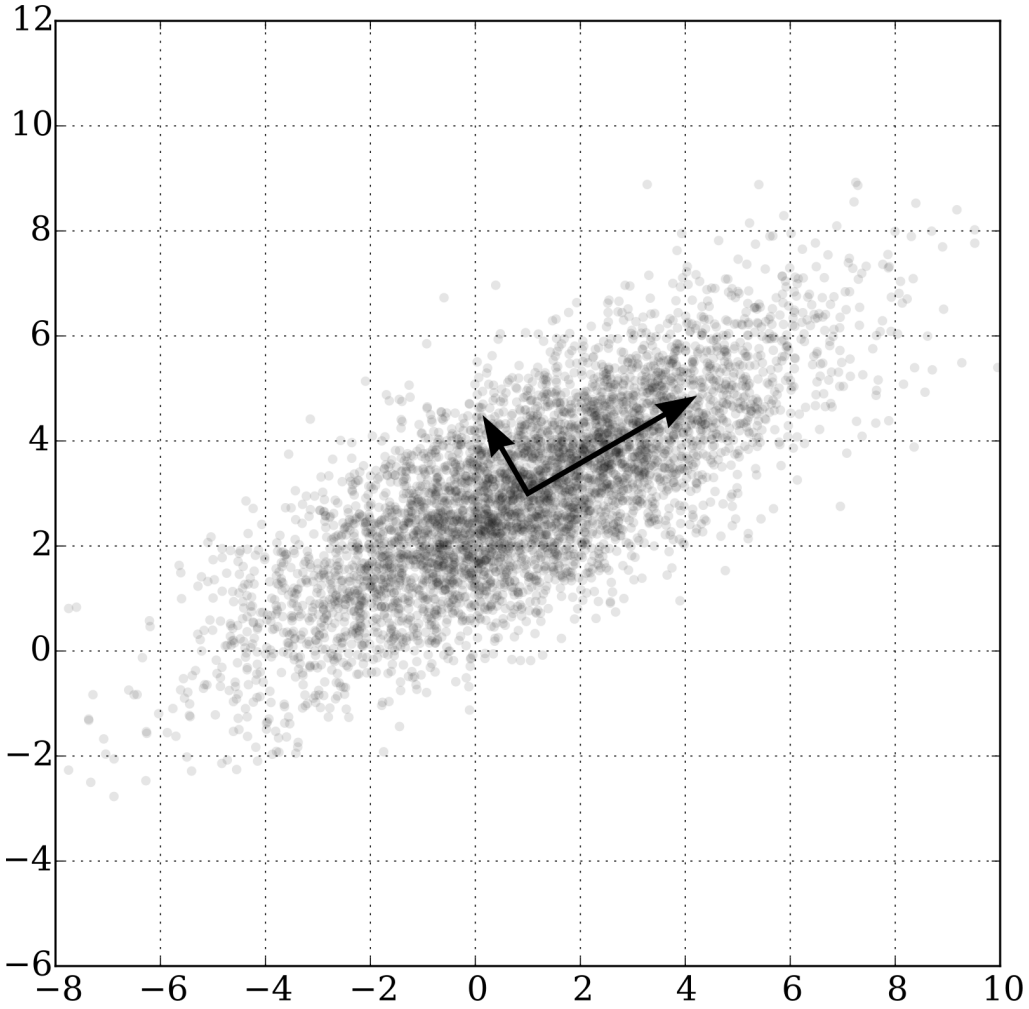
Формальная постановка

Найти P ортогональных векторов $w_1, \ldots, w_P \in \mathbb{R}^D$ единичной нормы $||w_p|| = 1$, так, чтобы для проекции

$$\mathcal{P}x = (x, w_1)w_1 + \cdots + (x, w_P)w_P$$

было минимально $\frac{1}{N}\sum\limits_{n=1}^{N} ||\mathcal{P}x_n - x_n||^2$.

Обыкновенный PCA: иллюстрация



Стрелки показывают направления векторов w_1 (длинная стрелка) и w_2 (короткая стрелка) для изображенного облака точек.

Обыкновенный PCA: вычисление $\{w_p\}_{p=1}^{P}$

Пусть для простоты $P = 1$: проектируем на 1D пространство. Требуется решить оптимизационную задачу: найти

$$w_1 = \arg\min_{||w_1||=1} \frac{1}{N} \sum_{n=1}^{N} ||(x_n, w_1)w_1 - x_n||^2$$

Распишем

$$\frac{1}{N} \sum_{n=1}^{N} ||(x_n, w_1)w_1 - x_n||^2 = \frac{1}{N} \sum_{n=1}^{N} \Big((x_n, w_1)^2 - 2(x_n, w_1)^2 + ||x_n||^2 \Big)$$

$$= \frac{1}{N} \sum_{n=1}^{N} \Big(||x_n||^2 - (x_n, w_1)^2 \Big)$$

Значит

$$w_1 = \arg\max_{||w_1||=1} \frac{1}{N} \sum_{n=1}^{N} (x_n, w_1)^2$$

Обыкновенный PCA: вычисление $\{w_p\}_{p=1}^P$

Поняли, что для того, чтобы найти оптимальный 1D проектор, нужно решить оптимизационную задачу

$$w_1 = \arg\max_{||w_1||=1} \frac{1}{N} \sum_{n=1}^{N} (x_n, w_1)^2$$

В правой части - выборочная дисперсия выборки $\{(x_n, w_1)\}_{n=1}^N$, нам ее нужно максимизировать.

Пусть X - матрица размера $N \times D$ из данных $x_1, \ldots, x_N$, тогда

$$\frac{1}{N} \sum_{n=1}^{N} (x_n, w_1)^2 = \frac{1}{N}(Xw_1, Xw_1) = w_1^T \frac{X^T X}{N} w_1 =: w_1^T C w_1,$$

где $C = \frac{X^T X}{N} = \frac{1}{N} \sum\limits_{n=1}^{N} x_n x_n^T$ - матрица $D \times D$ выборочных корреляций выборки $\{x_n\}_{n=1}^N$.
Упражнение: матрица C имеет ранг не больше N.

Обыкновенный PCA: вычисление $\{w_p\}_{p=1}^{P}$

Получили еще одну переформулировку оптимизационной задачи:

$$w_1 = \arg\max_{||w_1||=1} w_1^T C w_1$$

Она решается методом множителей Лагранжа: пусть

$$L(w_1, \lambda) = w_1^T C w_1 - \lambda(w_1^T w_1 - 1),$$

оптимальный w_1 должен удовлетворять (для какого-то $\lambda \in \mathbb{R}$) соотношениям

$$\frac{\partial L}{\partial w_1} = 2Cw_1 - 2\lambda w_1 = 0 \quad \text{значит} \quad Cw_1 = \lambda w_1$$

$$\frac{\partial L}{\partial \lambda} = w_1^T w_1 - 1 = 0 \qquad \text{значит} \quad ||w_1|| = 1$$

То есть w_1 - собственный вектор C.

Причем w_1 соответствует максимальному собственному числу.

Обыкновенный PCA: вычисление $\{w_p\}_{p=1}^P$

Абсолютно аналогичное вычисление можно провести и для общего случая проекции на P-мерное подпространство.

Оказывается, что w_p — собственные векторы матрицы выборочных корреляций C:

$$Cw_p = \lambda_p w_p, \qquad \text{где} \qquad \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_P.$$

Итог

Вектора $w_1, \ldots, w_P \in \mathbb{R}^D$, определяющие оптимальный проектор

$$\mathcal{P}x = (x, w_1)w_1 + \cdots + (x, w_P)w_P$$

являются собственными векторами матрицы $C = \frac{1}{N}\sum\limits_{n=1}^{N} x_n x_n^T$, соответствующими P максимальным собственным числам.

Обыкновенный PCA: итог

Итог

Вектора $w_1, \ldots, w_P \in \mathbb{R}^D$, определяющие оптимальный проектор

$$\mathcal{P}x = (x, w_1)w_1 + \cdots + (x, w_P)w_P$$

являются собственными векторами матрицы $C = \frac{1}{N}\sum\limits_{n=1}^{N} x_n x_n^T$, соответствующими P максимальным собственным числам.

Замечания:

- Данные: N векторов $x_1, x_2, \ldots, x_N$ размерности D.
  Цель: Найти линейное многообразие размерности P, на которое $\{x_i\}_{i=1}^N$ проектируются оптимально.
- Многообразие явно задано и вложено в пространство $\mathbb{R}^D$
- Мы умеем эффективно проектировать на него новые точки и считать расстояние до этого многообразия.

Обыкновенный PCA: итог

Итог

Вектора $w_1, \ldots, w_P \in \mathbb{R}^D$, определяющие оптимальный проектор

$$\mathcal{P}x = (x, w_1)w_1 + \cdots + (x, w_P)w_P$$

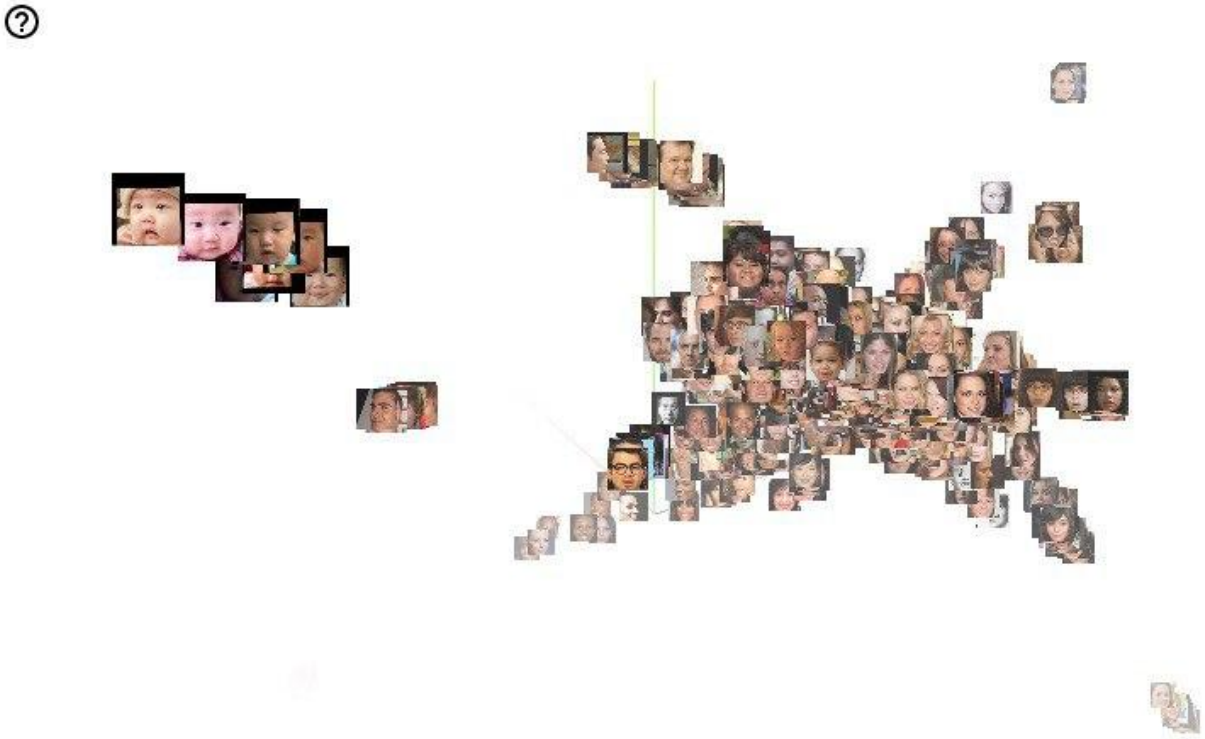
являются собственными векторами матрицы $C = \frac{1}{N}\sum\limits_{n=1}^{N} x_n x_n^T$, соответствующими P максимальным собственным числам.

Замечания (продолжение):

- Вычислительная сложность $O(D^3 + ND^2)$.
- Все определяется матрицей C, а матрица C определяется лишь скалярными произведениями — это будет основой для обобщения на ядерный случай.
- Аббревиатура SVD (Singular Value Decomposition) здесь используется как синоним диагонализации матрицы.

Еще применения PCA

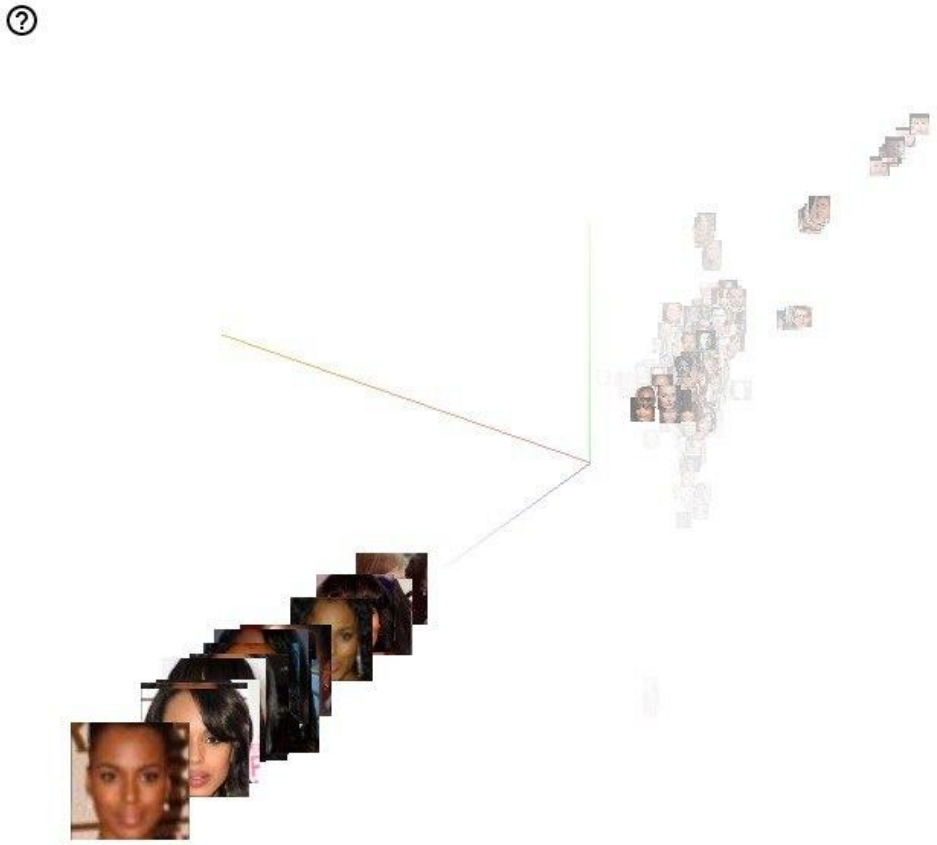
Некоторые примеры из нашего Manifold Learning проекта.



PCA 512-мерных эмбеддингов, построенных нейронной сетью по фотографиям человечески лиц

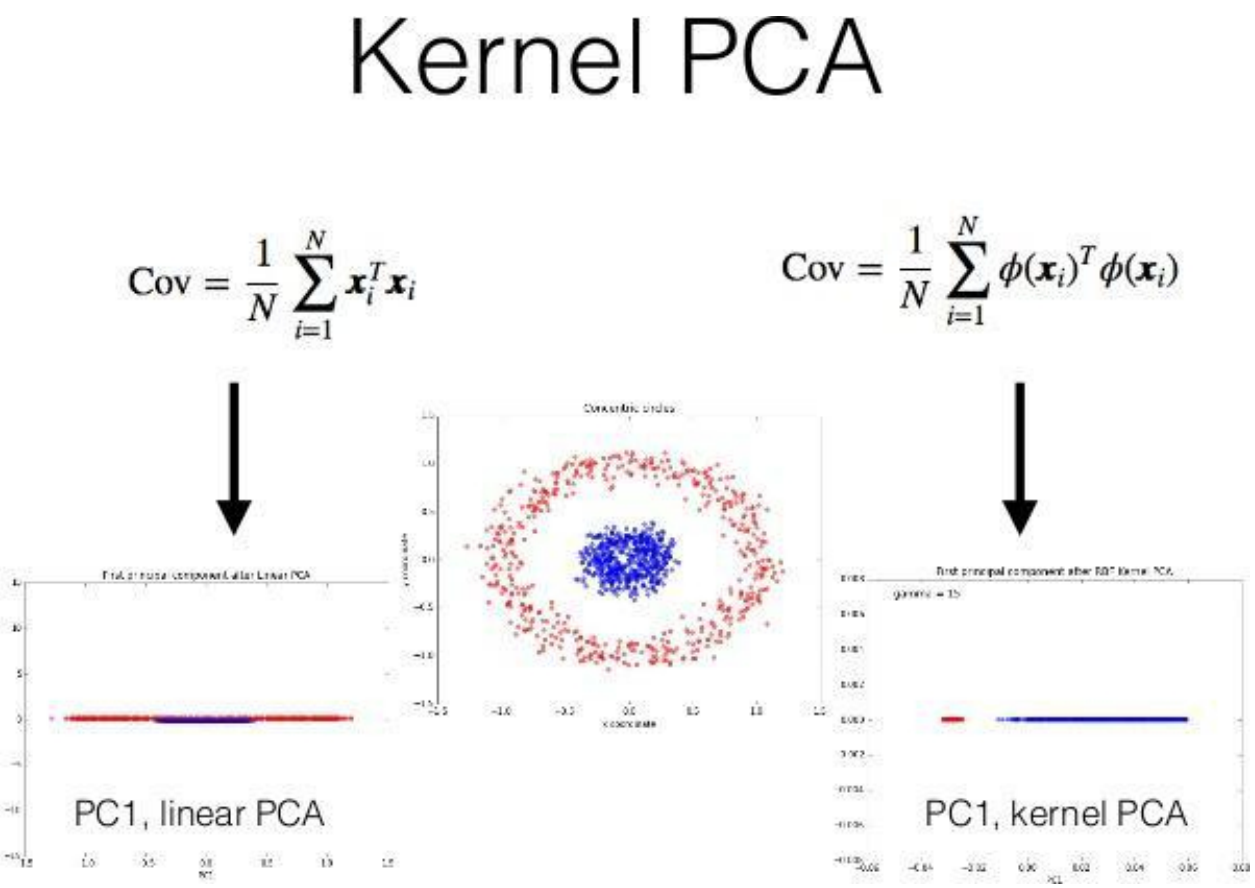
Еще применения PCA

Некоторые примеры из нашего Manifold Learning проекта.



PCA 512-мерных эмбеддингов, построенных нейронной сетью по фотографиям человечески лиц

Сравнение PCA и Kernel PCA на примере



Проекции на одномерное пространство с помощью PCA (слева) и Kernel PCA (справа). Первый перемешал два класса, второй — нет. Так будет далеко не всегда.

Kernel PCA: постановка задачи

Данные: N векторов $x_1, x_2, \ldots, x_N$ размерности D.

Данные: (Нелинейное) преобразование $\phi : \mathbb{R}^D \to \mathbb{H}$ в гильбертово пространство $\mathbb{H}$,

или

функция (ядро) $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ такая, что $k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathbb{H}}$.

Цель: Найти линейное подмногообразие в $\mathbb{H}$ разм-ти P, на которое $\{\phi(x_i)\}_{i=1}^N$ проектируются оптимально.

Утверждение

По произвольной функции ϕ можно построить ядро

$$k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathbb{H}}.$$

Функция k будет положительно определенной.

Kernel PCA: постановка задачи

Данные: N векторов $x_1, x_2, \ldots, x_N$ размерности D.

Данные: (Нелинейное) преобразование $\phi : \mathbb{R}^D \to \mathbb{H}$ в гильбертово пространство $\mathbb{H}$,

или

функция (ядро) $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ такая, что $k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathbb{H}}$.

Цель: Найти линейное подмногообразие в $\mathbb{H}$ разм-ти P, на которое $\{\phi(x_i)\}_{i=1}^N$ проектируются оптимально.

Утверждение (Moore–Aronszajn theorem)

По произвольной положительно определенной функции k можно построить гильбертово пространство $\mathbb{H}$ и (нелинейное) преобразование ϕ так, что

$$k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathbb{H}}.$$

Kernel PCA: постановка задачи

Данные: N векторов $x_1, x_2, \ldots, x_N$ размерности D.

Данные: (Нелинейное) преобразование $\phi : \mathbb{R}^d \to \mathbb{H}$ в гильбертово пространство $\mathbb{H}$,

или

функция (ядро) $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ такая, что $k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathbb{H}}$.

Цель: Найти линейное подмногообразие в $\mathbb{H}$ разм-ти P, на которое $\{\phi(x_i)\}_{i=1}^N$ проектируются оптимально.

Между ϕ и k нет взаимно однозначного соответствия.

Тем не менее, в большинстве задач ϕ и k взаимно заменяемы.

Замечание: зато есть взаимно-однозначное соответствие между положительно определенными функциями и интересным классом гильбертовых пространств (Reproducing Kernel Hilbert Spaces, RKHS).

Kernel PCA: поиск многообразия

Отвлечемся от обсуждения ядер и разберемся с Kernel PCA.

Пусть $\mathbb{H} = \mathbb{R}^H$, где $H \gg D$.
Не хотим сейчас возиться с бесконечномерным случаем.

Наивный подход

1. Вычисляем $\{\phi(x_n)\}_{n=1}^N$.
2. Используем обыкновенный PCA на данных $\{\phi(x_n)\}_{n=1}^N$.

Проблемы наивного подхода:

- Чаще всего ϕ нам не дано (дано только k).
- Теорема Moore–Aronszajn конструктивна, но обычно дает неприменимое на практике ϕ.
- Нужно будет работать с матрицей $H \times H$.

Kernel PCA: поиск многообразия

Вообразим все-таки, что $\{\phi(x_n)\}_{n=1}^N$ у нас есть.

Пусть Φ — матрица размера $N \times H$ из векторов $\{\phi(x_n)\}_{n=1}^N$.

Обыкновенный PCA предлагает сформировать матрицу

$$C = \frac{1}{N}\Phi^T\Phi = \frac{1}{N}\sum_{n=1}^{N}\phi(x_n)\phi(x_n)^T$$

размера $H \times H$ и решать задачу

$$Cw_p = \lambda_p w_p, \qquad p = 1, 2, \ldots, P.$$

Идея: несмотря на большой размер, матрица C имеет небольшой ранг (не больше N) — все должно быть проще.

Kernel PCA: поиск многообразия

Трюк: подставим выражение $C = \frac{1}{N}\sum\limits_{n=1}^{N}\phi(x_n)\phi(x_n)^T$ в

$$Cw_p = \lambda_p w_p, \qquad p = 1, 2, \ldots, P.$$

Получим

$$\frac{1}{N}\sum_{n=1}^{N}\phi(x_n)\langle\phi(x_n), w_p\rangle_{\mathbb{H}} = \lambda_p w_p$$

То есть собственные вектора являются линейными комбинациями векторов $\{\phi(x_n)\}_{n=1}^{N}$!

Kernel PCA: поиск многообразия

Продолжаем трюк.

Если предположить теперь, что $w_p = \sum\limits_{m=1}^{N} \alpha_{p,m}\phi(x_m)$, и подставить это выражение в

$$\frac{1}{N}\sum_{n=1}^{N}\phi(x_n)\langle\phi(x_n), w_p\rangle_{\mathbb{H}} = \lambda_p w_p,$$

выйдет

$$\frac{1}{N}\sum_{n=1}^{N}\sum_{m=1}^{N}\phi(x_n)\phi(x_n)^T\phi(x_m)\alpha_{p,m} = \lambda_p\sum_{m=1}^{N}\alpha_{p,m}\phi(x_m),$$

или

$$\frac{1}{N}\Phi^T\Phi\Phi^T\alpha_p = \lambda_p\Phi^T\alpha_p, \quad \text{то есть} \quad \Phi^T(\Phi\Phi^T\alpha_p - N\lambda_p\alpha_p) = 0$$

Это выполнено, если α_p — решение уравнения

$$K\alpha_p = N\lambda_p\alpha_p, \text{ где } K = \Phi\Phi^T \text{ — матрица } N \times N.$$

Kernel PCA: поиск многообразия

Заканчиваем трюк.

Получается, что для нахождения w_p достаточно решить задачу

$$K\alpha_p = N\lambda_p\alpha_p,$$

где $K = \Phi\Phi^T$ — матрица $N \times N$, причем $K_{nm} = k(x_n, x_m)$.

Важно: проекции на w_p можно вычислить даже не зная ϕ:

$$\langle \phi(x), w_p \rangle_{\mathbb{H}} = \sum_{m=1}^{N} \alpha_{p,m} \phi(x)^T \phi(x_m) = \sum_{m=1}^{N} \alpha_{p,m} k(x, x_m).$$

То же верно и для расстояния между $\phi(x)$ и его проекцией на вектора $\{w_p\}_{p=1}^{P}$. Пусть $k_{xn} = (k(x, x_1), \cdots, k(x, x_N))$, тогда

$$\left\| \phi(x) - \sum_{p=1}^{P} \langle \phi(x), w_p \rangle w_p \right\|_{\mathbb{H}}^2 = k(x, x) - k_{xn} \left(\sum_{p=1}^{P} \alpha_p \alpha_p^T \right) k_{xn}^T$$

Kernel PCA: центрирование

Если $\frac{1}{N}\sum\limits_{n=1}^{N}\phi(x_n) \neq 0$, вместо ϕ рассматривают

$$\tilde{\phi}(x) = \phi(x) - \frac{1}{N}\sum_{n=1}^{N}\phi(x_n),$$

а вместо k соответственно $\tilde{k}(x,y) = \left\langle \tilde{\phi}(x), \tilde{\phi}(y) \right\rangle$, то есть

$$\tilde{k}(x,y) = k(x,y) - \frac{1}{N}\sum_{n=1}^{N}(k(x,x_n) + k(x_n,y)) + \frac{1}{N^2}\sum_{n=1}^{N}\sum_{m=1}^{N}k(x_n,x_m).$$

В таком случае матрица K превращается в
$\widehat{K} = (I - \frac{ee^T}{N})K(I - \frac{ee^T}{N})$, где $e = (1,\ldots,1)^T$.

Kernel PCA: итог

Итог

По данным $x_1, \ldots, x_N$ и ядру k мы научились "воображать"
— неизвестное нам P-мерное линейное подпространство,
— натянутое на неизвестные нам вектора $w_1, \ldots, w_P$
— неизвестного нам гильбертова пространства $\mathbb{H}$.

На практике мы научились проектировать образ $\phi(x) \in \mathbb{H}$ любой точки $x \in \mathbb{R}^D$ на это P-мерное линейное подпространство:

$$\langle \phi(x), w_p \rangle_{\mathbb{H}} = \sum_{m=1}^{N} \alpha_{p,m} k(x, x_m).$$

и вычислять расстояние

$$||\phi(x) - \mathcal{P}\phi(x)||_{\mathbb{H}} = k(x, x) - k_{xn} \left(\sum_{p=1}^{P} \alpha_p \alpha_p^T \right) k_{xn}^T,$$

которое, в реальности, ограниченно применимо.

Kernel PCA: итог

Замечания

1. Вычислительная сложность $O(N^3)$.
2. Проблема выбора правильного ядра распадается на две:
   - выбор параметрического семейства, например $k_l(x, y) = e^{-\frac{(x-y)^2}{2l^2}}$.
   - выбор оптимальных параметров, например l в k_l.

   обе сложные.
3. Получить $x' \in \mathbb{R}^{\mathbb{D}}$, соответствующий $\mathcal{P}\phi(x)$ не так-то просто (x' называется "pre-image"), но для этого есть свои методы. Базовая идея – оптимизировать $||\phi(x') - \mathcal{P}\phi(x)||_{\mathbb{H}}$.
4. Kernel PCA, при правильном выборе ядра, можно превратить, например, в PCA, Isomap, Laplacian eigenmap, locally-linear embedding (LLE), ...

Еще немного про ядра

Давайте посмотрим на RBF (Gaussian/Squared Exponential)

$$k_l(x, y) = e^{-\frac{(x-y)^2}{2l^2}}$$

Приведем пример ϕ для k_l.

Утверждение

Пусть $\phi(x) : \mathbb{R} \to L^2(\mathbb{R})$ задана $\phi(x) = \left(u \to \frac{(2l)^{1/2}}{\pi^{1/4}} e^{-2l^2(x-u)^2}\right)$. Тогда $k_l(x, y) = \langle \phi(x), \phi(y) \rangle_{L^2(\mathbb{R})}$.

Это ядро часто критикуют за то, что матрица K получается плохо обусловленной (ее численная диагонализация сложна и дает неточный результат). Альтернатива: ядра Матерна

$$C_{\nu,\rho,\sigma}(x, y) = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu} \frac{|x-y|}{\rho}\right)^{\nu} K_\nu \left(\sqrt{2\nu} \frac{|x-y|}{\rho}\right).$$

Полезные ссылки

- Пример применения обыкновенного PCA к фотографиям лиц (получающиеся собственные вектора, и реконструкция из компонент, которые сохраняют 80% дисперсии):

  https://medium.com/@sebastiannorena/

  pca-principal-components-analysis-applied-to-images-of-faces-d2fc2c083371.
- Два примера работы Kernel PCA с разными ядрами:

  https://en.wikipedia.org/wiki/Kernel_principal_component_analysis.
- Про “ядерный” SVM против нейронных сетей:

  http://yann.lecun.com/exdb/mnist/.
- Проекция многообразия шрифтов на 2D с помощью “ядерных” гауссовских процессов (модель GP-LVM):

  http://vecg.cs.ucl.ac.uk/Projects/projects_fonts/projects_fonts.html.