LDA 数学八卦

Rickjin(靳志辉), version 1.0

2013 年 2 月 8 日

Contents

0.1 开篇

在Machine Learning 中，LDA 是两个常用模型的简称：Linear Discriminant Analysis 和Latent Dirichlet Allocation，在这篇文章中我们主要八卦的是后者。LDA 是一个在文本建模中很著名的模型，类似于SVD, PLSA 等模型，可以用于浅层语义分析,在文本语义分析中是一个很有用的模型。很不幸的是，这个模型中涉及的数学知识有点多，包括Gamma 函数，Dirichlet 分布，Dirichlet-Multinomial 共轭，Gibbs Sampling, Variational Inference, 贝叶斯文本建模，PLSA 建模, 以及LDA 文本建模。

这篇文章的主要目标，就是科普在学习理解LDA 模型中，需要了解的一些重要的数学知识。预设的读者是做自然语言处理、机器学习、数据挖掘方向的工程师，要读懂这篇科普，需要的数学基础知识基本上不超过陈希孺先生的《概率论与数理统计》这本书。

文章标题挂上“八卦”两字，因为八卦意味着自由、不拘束、可以天马行空，细节处理上也难免有不严谨的地方；当然我也希望八卦是相对容易理解的，即便他是关于数学的八卦。对于本文中的任何评论，欢迎发信到我的新浪微博帐号rickjin, 或者是邮箱zhihuijin@gmail.com。

0.2 神奇的Gamma函数

0.2.1 Gamma 函数诞生记

学高等数学的时候，我们都学习过如下一个长相有点奇特的Gamma函数

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} dt$$

通过分部积分的方法，可以推导出这个函数有如下的递归性质

$$\Gamma(x+1) = x\Gamma(x)$$

于是很容易证明，$\Gamma(x)$ 函数可以当成是阶乘在实数集上的延拓，具有如下性质

$$\Gamma(n) = (n-1)!$$

学习了Gamma 函数之后，多年以来我一直有两个疑问：

1. 这个长得这么怪异的一个函数，数学家是如何找到的；
2. 为何定义Γ 函数的时候，不使得这个函数的定义满足$\Gamma(n) = n!$ 而是$\Gamma(n) = (n-1)!$

最近翻了一些资料，发现有不少文献资料介绍Gamma 函数发现的历史，要说清楚它需要一定的数学推导，这儿只是简要的说一些主线。

1728年，哥德巴赫在考虑数列插值的问题，通俗的说就是把数列的通项公式定义从整数集合延拓到实数集合，例如数列$1, 4, 9, 16, \cdots$ 可以用通项公式n^2自然的表达，即便n 为实数的时候，这个通项公式也是良好定义的。直观的说也就是可以找到一条平滑的曲线通过$y = x^2$通过所有的整数点(n, n^2)这些点，从而可以把定义在整数集上的公式延拓到实数集合。一天哥德巴赫开始处理阶乘序列$1, 2, 6, 24, 120, 720, \cdots$, 我们可以计算$2!, 3!$, 是否可以计算$2.5!$呢？我们把最初的一些$(n, n!)$的点画在坐标轴上，确实可以看到，容易画出一条通过这些点的平滑曲线

THE FACTORIALS

n:	1	2	3	4	5	6	7	8	$\cdots$
$n!$:	1	2	6	24	120	720	5040	40,320	$\cdots$

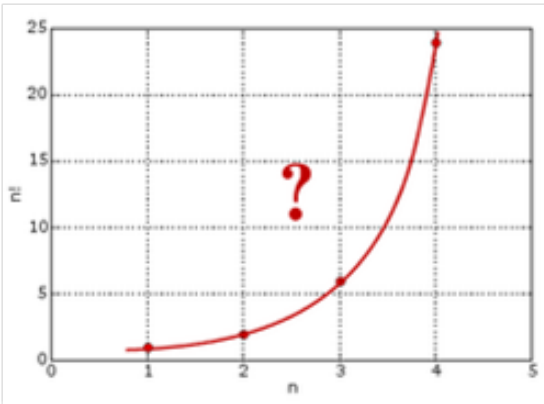


Figure 1: 通过$(n, n!)$的曲线

哥德巴赫无法解决这个问题，于是写信请教尼古拉斯.贝努利和他的弟弟丹尼尔.贝努利，由于欧拉当时和丹尼尔.贝努利在一块，他也因此得知了这个问题。而欧拉于1729 年完美的解决了这个问题，由此导致了Γ 函数的诞生，当时欧拉只有22岁。

事实上首先解决$n!$的插值计算问题的是丹尼尔.贝努利，他发现，如果m, n都是正整数，如果$m \to \infty$，有

$$\frac{1 \cdot 2 \cdot 3 \cdots m}{(1+n)(2+n)\cdots(m-1+n)}(m+\frac{n}{2})^{n-1} \to n!$$

于是用这个无穷乘积的方式可以把$n!$的定义延拓到实数集合。例如，取$n = 2.5$,

m 足够大，基于上式就可以近似计算出2.5!。

欧拉也偶然的发现$n!$ 可以用如下的一个无穷乘积表达

$$\left[\left(\frac{2}{1}\right)^n \frac{1}{n+1}\right]\left[\left(\frac{3}{2}\right)^n \frac{2}{n+2}\right]\left[\left(\frac{4}{3}\right)^n \frac{3}{n+3}\right]\cdots = n! \tag{1}$$

用极限形式，这个式子整理后可以写为

$$\lim_{m\to\infty} \frac{1\cdot 2\cdot 3\cdots m}{(1+n)(2+n)\cdots(m+n)}(m+1)^n = n! \tag{2}$$

左边可以整理为

$$\begin{aligned}
&\frac{1\cdot 2\cdot 3\cdots m}{(1+n)(2+n)\cdots(m+n)}(m+1)^n \\
=&1\cdot 2\cdot 3\cdots n\cdot \frac{(n+1)(n+2)m}{(1+n)(2+n)\cdots m}\cdot \frac{(m+1)^n}{(m+1)(m+2)\cdots(m+n)} \\
=&n!\frac{(m+1)^n}{(m+1)(m+2)\cdots(m+n)} \\
=&n!\prod_{k=1}^{n}\frac{m+1}{m+k} \to n! \qquad (m\to\infty)
\end{aligned}$$

所以(1)、(2)式都成立。

欧拉开始尝试从一些简单的例子开始做一些计算，看看是否有规律可循，欧拉极其擅长数学的观察与归纳。当$n = 1/2$ 的时候，带入(1)式计算，整理后可以得到

$$\left(\frac{1}{2}\right)! = \sqrt{\frac{2\cdot 4}{3\cdot 3}\cdot\frac{4\cdot 6}{5\cdot 5}\cdot\frac{6\cdot 8}{7\cdot 7}\cdot\frac{8\cdot 10}{9\cdot 9}\cdots}$$

然而右边正好和著名的Wallis 公式关联。Wallis 在1665年使用插值方法计算半圆曲线$y = \sqrt{x(1-x)}$ 下的面积(也就是直径为1的半圆面积)的时候，得到关于π的如下结果，

$$\frac{2\cdot 4}{3\cdot 3}\cdot\frac{4\cdot 6}{5\cdot 5}\cdot\frac{6\cdot 8}{7\cdot 7}\cdot\frac{8\cdot 10}{9\cdot 9}\cdots = \frac{\pi}{4}$$

于是，欧拉利用Wallis 公式得到了如下一个很漂亮的结果

$$\left(\frac{1}{2}\right)! = \frac{\sqrt{\pi}}{2}$$

欧拉和高斯都是具有超凡直觉的数学家，但是欧拉和高斯的风格迥异。高斯是个老狐狸，数学上非常严谨，发表结果的时候却都把思考的痕迹抹去，只留下漂亮的结果，这招致了一些数学家对高斯的批评；而欧拉的风格不同，经常通过经验直觉做大胆的猜测，而他的文章中往往留下他如何做数学猜想的痕迹，而文章有的时候论证不够严谨。拉普拉斯曾说过：”读读欧拉,他是所有人

Figure 2: 大数学家欧拉

的老师。”波利亚在他的名著《数学与猜想》中也对欧拉做数学归纳和猜想的方式推崇备至。

欧拉看到$(\frac{1}{2})!$ 中居然有π, 对数学家而言，有π 的地方必然有和圆相关的积分。由此欧拉猜测$n!$ 一定可以表达为某种积分形式，于是欧拉开始尝试把$n!$ 表达为积分形式。虽然Wallis 的时代微积分还没有发明出来，Wallis 是使用插值的方式做推导计算的，但是Wallis 公式的推导过程基本上就是在处理积分$\int_0^1 x^{\frac{1}{2}}(1-x)^{\frac{1}{2}}dx$，受Wallis 的启发，欧拉开始考虑如下的一般形式的积分

$$J(e,n)=\int_0^1 x^e(1-x)^n dx$$

此处n 为正整数，e 为正实数。利用分部积分方法，容易得到

$$J(e,n)=\frac{n}{e+1}J(e+1,n-1)$$

重复使用上述迭代公式，最终可以得到

$$J(e,n)=\frac{1\cdot 2\cdots n}{(e+1)(e+2)\cdots(e+n+1)}$$

于是欧拉得到如下一个重要的式子

$$n!=(e+1)(e+2)\cdots(e+n+1)\int_0^1 x^e(1-x)^n dx$$

接下来，欧拉使用了一点计算技巧，取$e=f/g$ 并且令$f\to 1, g\to 0$, 然后对上式右边计算极限(极限计算的过程此处略去，推导不难，有兴趣的同学看后面的参考文献吧)，于是欧拉得到如下简洁漂亮的结果：

$$n!=\int_0^1(-\log t)^n dt$$

欧拉成功的把$n!$表达为了积分形式！如果我们做一个变换$t = e^{-u}$,就可以得到我们常见的Gamma 函数形式

$$n! = \int_0^\infty u^n e^{-u} du$$

于是,利用上式把阶乘延拓到实数集上，我们就得到Gamma 函数的一般形式

$$\Gamma(x) = \int_0^1 (-\log t)^{x-1} dt = \int_0^\infty t^{x-1} e^{-t} dt$$

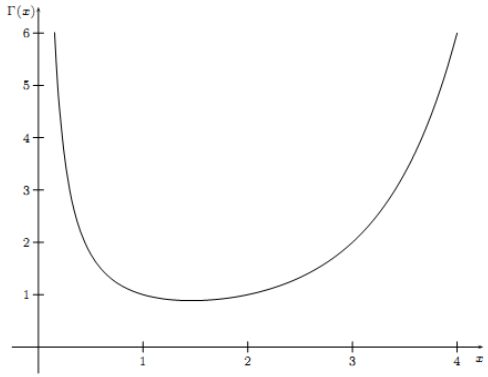


Figure 3: $\Gamma(x)$

Gamma 函数找到了，我们来看看第二个问题，为何Gamma 函数被定义为$\Gamma(n) = (n-1)!$, 这看起来挺别扭的。如果我们稍微修正一下，把Gamma 函数定义中的t^{x-1} 替换为t^x

$$\Gamma(x) = \int_0^\infty t^x e^{-t} dt$$

这不就可以使得$\Gamma(n) = n!$了嘛。欧拉最早的Gamma函数定义还真是如上所示，选择了$\Gamma(n) = n!$，可是欧拉不知出于什么原因，后续修改了Gamma 函数的定义，使得$\Gamma(n) = (n-1)!$。而随后勒让德等数学家对Gamma 函数的进一步深入研究中，认可了这个定义，于是这个定义就成为了既成事实。有数学家猜测，一个可能的原因是欧拉研究了如下积分

$$B(m,n) = \int_0^1 x^{m-1}(1-x)^{n-1} dx$$

这个函数现在称为Beta 函数。如果Gamma 函数的定义选取满足$\Gamma(n) = (n-1)!$, 那么有

$$B(m,n) = \frac{\Gamma(m)\Gamma(n)}{\Gamma(m+n)}$$

非常漂亮的对称形式。可是如果选取$\Gamma(n) = n!$ 的定义，令

$$E(m,n) = \int_0^1 x^m (1-x)^n dx$$

则有

$$E(m,n) = \frac{\Gamma(m)\Gamma(n)}{\Gamma(m+n+1)}$$

这个形式显然不如$B(m,n)$优美，而数学家总是很在乎数学公式的美感的。

要了解更多的Gamma 函数的历史，推荐阅读

- Philip J. Davis, Leonhard Euler's Integral: A Historical Profile of the Gamma Function
- Jacques Dutka, The Early History of the Factorial Function
- Detlef Gronnau, Why is the gamma function so as it is?

0.2.2 Gamma 函数欣赏

> *Each generation has found something of interest to say about the gamma function. Perhaps the next generation will also.*
> *—Philip J.Davis*

Gamma 函数从它诞生开始就被许多数学家进行研究，包括高斯、勒让德、威尔斯特拉斯、柳维尔等等。这个函数在现代数学分析中被深入研究，在概率论中也是无处不在，很多统计分布都和这个函数相关。Gamma 函数作为阶乘的推广，首先它也有和Stirling 公式类似的一个结论

$$\Gamma(x) \sim \sqrt{2\pi} e^{-x} x^{x-\frac{1}{2}}$$

另外，Gamma 函数不仅可以定义在实数集上，还可以延拓到整个复平面上。

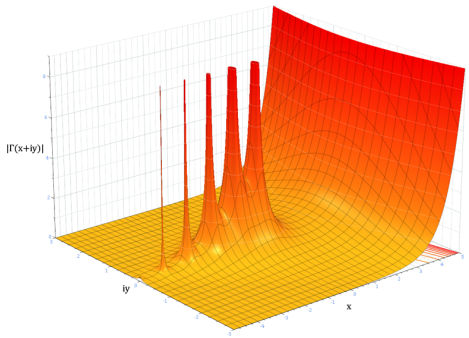


Figure 4: 复平面上的Gamma 函数

Gamma 函数有很多妙用，它不但使得(1/2)! 的计算有意义，还能扩展很多其他的数学概念。比如导数，我们原来只能定义一阶、二阶等整数阶导数，有了Gamma 函数我们可以把函数导数的定义延拓到实数集，从而可以计算1/2阶导数,同样的积分作为导数的逆运算也可以有分数阶。我们先考虑一下x^n 的各阶导数

first derivative	nx^{n-1},
second derivative	$n(n-1)x^{n-2}$,
third derivative	$n(n-1)(n-2)x^{n-3}$,
⋯	
k-th derivative	$n(n-1)(n-2)\cdots(n-k+1)x^{n-k}=\frac{n!}{(n-k)!}x^{n-k}$,

Figure 5: x^n 的各阶导数

由于k阶导数可以用阶乘表达，于是我们用Gamma 函数表达为

$$\frac{\Gamma(n+1)}{\Gamma(n-k+1)}x^{n-k}$$

于是基于上式，我们可以把导数的阶从整数延拓到实数集。例如，取$n=1,k=\frac{1}{2}$ 我们可以计算x 的$\frac{1}{2}$阶导数为

$$\frac{\Gamma(1+1)}{\Gamma(1-1/2+1)}x^{1-1/2}=\frac{2\sqrt{x}}{\sqrt{\pi}}$$

很容易想到对于一般的函数$f(x)$ 通过Taylor 级数展开可以表达为幂级数，于是借用x^n 的分数阶导数，我们可以尝试定义出任意函数的分数阶导数。不过有点遗憾的是这种定义方法并非良定义的，不是对所有函数都适用，但是这个思想却是被数学家广泛采纳了，并由此发展了数学分析中的一个研究课题：Fractional Calculus, 在这种微积分中，分数阶的导数和积分都具有良定义，而这t都依赖于Gamma 函数。

Gamma 函数和欧拉常数γ 有密切关系，可以发现

$$\gamma=-\frac{d\Gamma(x)}{dx}|_{x=1}=\lim_{n\to\infty}(1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}-\log n)$$

进一步还可以发现Gamma 函数和黎曼函数$\zeta(s)$有密切联系，

$$\zeta(s)=1+\frac{1}{2^s}+\frac{1}{3^s}+\cdots$$

而ζ 函数涉及了数学中著名的黎曼猜想和素数的分布定理。希尔伯特曾说，如果他在沉睡1000年后醒来, 他将问的第一个问题便是:黎曼猜想得到证明了吗？

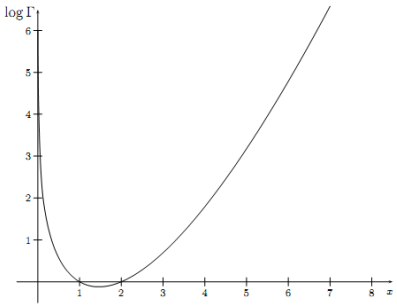


Figure 6: $\log\Gamma(x)$

从Gamma 函数的图像我们可以看到它是一个凸函数, 不仅如此, $\log\Gamma(x)$ 也是一个凸函数，数学上可以证明如下定理:

定理0.2.1 (Bohr-Mullerup)　如果$f:(0,\infty)\to(0,\infty)$,且满足

1. $f(1)=1$
2. $f(x+1)=xf(x)$
3. $\log f(x)$ 是凸函数

那么$f(x)=\Gamma(x)$, 也就是$\Gamma(x)$是唯一满足以上条件的函数。

如下函数被称为Digamma 函数，

$$\Psi(x)=\frac{d\log\Gamma(x)}{dx}$$

这也是一个很重要的函数，在涉及求Dirichlet 分布相关的参数的极大似然估计时，往往需要使用到这个函数。Digamma 函数具有如下一个漂亮的性质

$$\Psi(x+1)=\Psi(x)+\frac{1}{x}$$

函数$\Psi(x)$和欧拉常数γ 以及ζ 函数都有密切关系，令

$$\Psi_n(x)=\frac{d^{n+1}\log\Gamma(x)}{dx^{n+1}}$$

则$\Psi_0(x)=\Psi(x)$,可以证明

$$\Psi(1)=-\gamma,\Psi(2)=1-\gamma$$

$$\Psi_1(1) = \zeta(2) = \frac{\pi^2}{6}, \Psi_2(1) = -2\zeta(3)$$

所以Gamma 函数在数学上是很有魅力的，它在数学上应用广泛，不仅能够被一个理科本科生很好的理解，本身又足够的深刻，具有很多漂亮的数学性质，历史上吸引了众多一流的数学家对它进行研究。美国数学家Philip J.Davis 写了篇很有名的介绍Gamma 函数的文章：“Leonhard Euler's Integral: A Historical Profile of the Gamma Function”，文中对Gamma 函数一些特性发现的历史进行了很详细的描述，这篇文章获得了Chauvenet Prize(美国数学会颁发的数学科普最高奖)。

0.2.3 从二项分布到Gamma 分布

Gamma 函数在概率统计中频繁现身，众多的统计分布，包括常见的统计学三大分布(t 分布，χ^2 分布，F 分布)、Beta分布、Dirichlet 分布的密度公式中都有Gamma 函数的身影；当然发生最直接联系的概率分布是直接由Gamma 函数变换得到的Gamma 分布。对Gamma 函数的定义做一个变形，就可以得到如下式子

$$\int_0^\infty \frac{x^{\alpha-1}e^{-x}}{\Gamma(\alpha)}dx = 1$$

于是，取积分中的函数作为概率密度，就得到一个形式最简单的Gamma 分布的密度函数

$$Gamma(x|\alpha) = \frac{x^{\alpha-1}e^{-x}}{\Gamma(\alpha)}$$

如果做一个变换$x = \beta t$, 就得到Gamma 分布的更一般的形式

$$Gamma(t|\alpha,\beta) = \frac{\beta^\alpha t^{\alpha-1}e^{-\beta t}}{\Gamma(\alpha)}$$

其中α 称为shape parameter, 主要决定了分布曲线的形状;而β 称为rate parameter 或者inverse scale parameter ($\frac{1}{\beta}$ 称为scale parameter),主要决定曲线有多陡。

Gamma 分布在概率统计领域也是一个万人迷，众多统计分布和它有密切关系。指数分布和χ^2 分布都是特殊的Gamma 分布。另外Gamma 分布作为先验分布是很强大的，在贝叶斯统计分析中被广泛的用作其它分布的先验。如果把统计分布中的共轭关系类比为人类生活中的情侣关系的话，那指数分布、Poission分布、正态分布、对数正态分布都可以是Gamma 分布的情人。接下来的内容中中我们主要关注$\beta = 1$的简单形式的Gamma 分布。

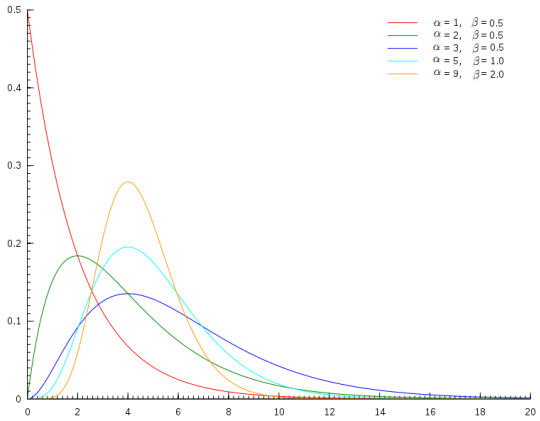


Figure 7: $Gamma(t|\alpha,\beta)$分布图像

Gamma 分布首先和Poisson 分布、Poisson 过程发生密切的联系。我们容易发现Gamma 分布的概率密度和Poisson 分布在数学形式上具有高度的一致性。参数为λ的Poisson 分布，概率写为

$$Poisson(X=k|\lambda)=\frac{\lambda^k e^{-\lambda}}{k!}$$

在Gamma 分布的密度中取$\alpha=k+1$ 得到

$$Gamma(x|\alpha=k+1)=\frac{x^k e^{-x}}{\Gamma(k+1)}=\frac{x^k e^{-x}}{k!}$$

所以这两个分布数学形式上是一致的，只是Poisson 分布是离散的，Gamma 分布是连续的，可以直观的认为Gamma 分布是Poisson 分布在正实数集上的连续化版本。

这种数学上的一致性是偶然的吗？这个问题我个人曾经思考了很久，终于想明白了从二项分布出发能把Gamma 分布和Poisson 分布紧密联系起来。我们在概率统计中都学过$Poisson(\lambda)$ 分布可以看成是二项分布$B(n,p)$在$np=\lambda, n\to\infty$ 条件下的极限分布。如果你对二项分布关注的足够多，可能会知道二项分布的随机变量$X\sim B(n,p)$满足如下一个很奇妙的恒等式

$$P(X\le k)=\frac{n!}{k!(n-k-1)!}\int_p^1 t^k(1-t)^{n-k-1}dt \tag{3}$$

这个等式反应的是二项分布和Beta 分布之间的关系，证明并不难，它可以用一个物理模型直观的做概率解释，而不需要使用复杂的数学分析的方法做证明。由于这个解释和Beta 分布有紧密的联系，所以这个直观的概率解释我

们放到下一个章节，讲解Beta/Dirichlet 分布的时候进行。此处我们暂时先承认(3)这个等式成立。我们在等式右侧做一个变换$t = \frac{x}{n}$,得到

$$\begin{aligned} P(X \leq k) &= \frac{n!}{k!(n-k-1)!} \int_p^1 t^k (1-t)^{n-k-1} dt \\ &= \frac{n!}{k!(n-k-1)!} \int_{np}^n (\frac{x}{n})^k (1-\frac{x}{n})^{n-k-1} d\frac{x}{n} \\ &= \frac{(n-1)!}{k!(n-k-1)!} \int_{np}^n (\frac{x}{n})^k (1-\frac{x}{n})^{n-k-1} dx \\ &= \int_{np}^n \binom{n-1}{k} (\frac{x}{n})^k (1-\frac{x}{n})^{n-k-1} dx \\ &= \int_{np}^n Binomial(Y = k|n-1, \frac{x}{n}) dx \end{aligned} \tag{4}$$

上式左侧是二项分布$B(n,p)$, 而右侧为无穷多个二项分布$B(n-1, \frac{x}{n})$的积分和, 所以可以写为

$$Binomial(X \leq k|n,p) = \int_{np}^n Binomial(Y = k|n-1, \frac{x}{n}) dx \tag{5}$$

实际上，对(5)式两边在条件$np = \lambda, n \to \infty$ 下取极限，则左边有$B(n,p) \to Poisson(\lambda)$, 而右边有$B(n-1, \frac{x}{n}) \to Poisson(x)$,所以得到

$$Poisson(X \leq k|\lambda) = \int_\lambda^\infty Poisson(Y = k|x) dx \tag{6}$$

把上式右边的Possion 分布展开，于是得到

$$Poisson(X \leq k|\lambda) = \int_\lambda^\infty Poisson(Y = k|x) dx = \int_\lambda^\infty \frac{x^k e^{-x}}{k!} dx$$

所以对于们得到如下一个重要而有趣的等式

$$Poisson(X \leq k|\lambda) = \int_\lambda^\infty \frac{x^k e^{-x}}{k!} dx \tag{7}$$

接下来我们继续玩点好玩的，对上边的等式两边在$\lambda \to 0$ 下取极限，左侧Poisson分布是要至少发生k个事件的概率，$\lambda \to 0$ 的时候就不可能有事件发生了，所以$P(X \leq k) \to 1$, 于是我们得到

$$1 = \lim_{\lambda \to 0} \int_\lambda^\infty \frac{x^k e^{-x}}{k!} dx = \int_0^\infty \frac{x^k e^{-x}}{k!} dx$$

在这个积分式子说明$f(x) = \frac{x^k e^{-x}}{k!}$在正实数集上是一个概率分布函数，而这个函数恰好就是Gamma 分布。我们继续把上式右边中的$k!$ 移到左边，于是得到

$$k! = \int_0^\infty x^k e^{-x} dx$$

于是我们得到了$k!$ 表示为积分的方法。

看，我们从二项分布的一个等式出发, 同时利用二项分布的极限是Possion分布这个性质，基于比较简单的逻辑，推导出了Gamma 分布，同时把$k!$ 表达为Gamma 函数了！实际上以上推导过程是给出了另外一种相对简单的发现Gamma 函数的途径。

回过头我们看看(7)式,非常有意思，它反应了Possion 分布和Gamma 分布的关系，这个和(3)式中中反应的二项分布和Beta 分布的关系具有完全相同的结构。把(7)式变形一下得到

$$Poisson(X \leq k|\lambda) + \int_0^{\lambda} \frac{x^k e^{-x}}{k!} dx = 1$$

我们可以看到，Poisson分布的概率累积函数和Gamma 分布的概率累积函数有互补的关系。

其实(3)和(7)这两个式子都是陈希儒院士的《概率论与数理统计》这本书第二章的课后习题，不过陈老师习题答案中给的证明思路是纯粹数学分析的证明方法，虽然能证明等式成立，但是看完证明后无法明白这两个等式是如何被发现的。上诉的论述过程说明，从二项分布出发，这两个等式都有可以很好的从概率角度进行理解。希望以上的推导过程能给大家带来一些对Gamma 函数和Gamma 分布的新的理解，让Gamma 分布不再神秘。((7)式是初等概率论的一个结论，俗称Possion-Gamma duality, 也可以从Possion 分布进行推导，不过这要求大家对Possion 过程有一定的认识。)

0.3 认识Beta/Dirichlet分布

0.3.1 撒旦的游戏—认识Beta 分布

统计学就是猜测上帝的游戏,当然我们不总是有机会猜测上帝，运气不好的时候就得揣度魔鬼的心思。有一天你被魔鬼撒旦抓走了，撒旦说：“你们人类很聪明，而我是很仁慈的，和你玩一个游戏，赢了就可以走，否则把灵魂出卖给我。游戏的规则很简单，我有一个魔盒，上面有一个按钮，你每按一下按钮，就均匀的输出一个[0,1]之间的随机数，我现在按10下，我手上有10个数，你猜第7大的数是什么，偏离不超过0.01就算对。”你应该怎么猜呢？

从数学的角度抽象一下，上面这个游戏描述如下

Algorithm 1 游戏1

1: $X_1, X_2, \cdots, X_n \overset{\text{iid}}{\sim} Uniform(0,1)$，
2: 把这n 个随机变量排序后得到顺序统计量$X_{(1)}, X_{(2)} \cdots, X_{(n)}$，
3: 问$X_{(k)}$ 的分布是什么

对于不喜欢数学的同学而言，估计每个概率分布都是一个恶魔，那在概率统计学中，均匀分布应该算得上是潘多拉魔盒，几乎所有重要的概率分布都可以从均匀分布$Uniform(0,1)$中生成出来。

Figure 8: 潘多拉魔盒

对于上面的游戏而言$n = 10, k = 7$, 如果我们能求出$X_{(7)}$ 的分布的概率密度，那么用概率密度的极值点去做猜测就是最好的策略。对于一般的情形，$X_{(k)}$ 的分布是什么呢？那我们尝试计算一下$X_{(k)}$ 落在一个区间$[x, x+\Delta x]$ 的概率，也就是求如下概率值

$$P(x \le X_{(k)} \le x + \Delta x) = ?$$

把[0,1] 区间分成三段$[0, x), [x, x+\Delta x], (x+\Delta x, 1]$, 我们先考虑简单的情形，假设$n$ 个数中只有一个落在了区间$[x, x+\Delta x]$内，则因为这个区间内的数$X_{(k)}$是第k大的，则$[0, x)$中应该有$k-1$ 个数，$(x, 1]$这个区间中应该有$n-k$ 个数。不

失一般性，我们先考虑如下一个符合上述要求的事件E

$$\begin{aligned} E = \{&X_1 \in [x, x+\Delta x], \\ &X_i \in [0, x) \quad (i = 2, \cdots, k), \\ &X_j \in (x+\Delta x, 1] \quad (j = k+1, \cdots, n)\} \end{aligned}$$

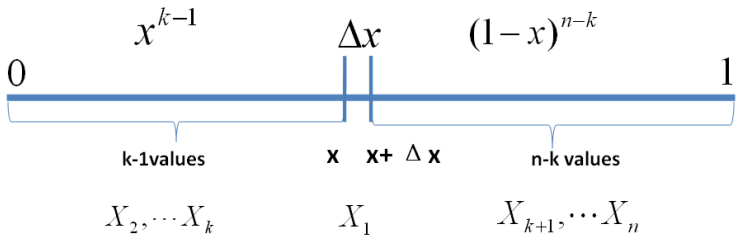


Figure 9: 事件E

则有

$$\begin{aligned} P(E) &= \prod_{i=1}^{n} P(X_i) \\ &= x^{k-1}(1-x-\Delta x)^{n-k}\Delta x \\ &= x^{k-1}(1-x)^{n-k}\Delta x + o(\Delta x) \end{aligned}$$

$o(\Delta x)$表示Δx的高阶无穷小。显然，由于不同的排列组合，即n个数中有一个落在$[x, x+\Delta x]$区间的有n种取法，余下$n-1$个数中有$k-1$个落在$[0, x)$的有$\binom{n-1}{k-1}$种组合，所以和事件E等价的事件一共有$n\binom{n-1}{k-1}$个。

继续考虑稍微复杂一点情形，假设n 个数中有两个数落在了区间$[x, x+\Delta x]$，

$$\begin{aligned} E' = \{&X_1, X_2 \in [x, x+\Delta x], \\ &X_i \in [0, x) \quad (i = 3, \cdots, k), \\ &X_j \in (x+\Delta x, 1] \quad (j = k+1, \cdots, n)\} \end{aligned}$$

则有

$$P(E') = x^{k-2}(1-x-\Delta x)^{n-k}(\Delta x)^2 = o(\Delta x)$$

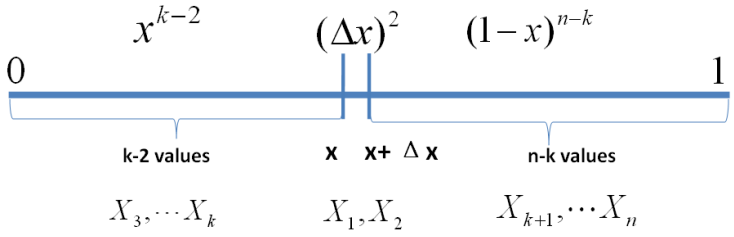


Figure 10: 事件E'

从以上分析我们很容易看出，只要落在$[x, x+\Delta x]$内的数字超过一个，则对应的事件的概率就是$o(\Delta x)$。于是

$$\begin{aligned} & P(x \le X_{(k)} \le x+\Delta x) \\ &= n\binom{n-1}{k-1} P(E) + o(\Delta x) \\ &= n\binom{n-1}{k-1} x^{k-1}(1-x)^{n-k}\Delta x + o(\Delta x) \end{aligned}$$

所以，可以得到$X_{(k)}$的概率密度函数为

$$\begin{aligned} f(x) &= \lim_{\Delta x \to 0} \frac{P(x \le X_{(k)} \le x+\Delta x)}{\Delta x} \\ &= n\binom{n-1}{k-1} x^{k-1}(1-x)^{n-k} \\ &= \frac{n!}{(k-1)!(n-k)!} x^{k-1}(1-x)^{n-k} \quad x \in [0,1] \end{aligned}$$

利用Gamma 函数，我们可以把$f(x)$ 表达为

$$f(x) = \frac{\Gamma(n+1)}{\Gamma(k)\Gamma(n-k+1)} x^{k-1}(1-x)^{n-k}$$

还记得神奇的Gamma 函数可以把很多数学概念从整数集合延拓到实数集合吧。我们在上式中取$\alpha = k, \beta = n-k+1$, 于是我们得到

$$f(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1} \tag{8}$$

这个就是一般意义上的Beta 分布！可以证明，在α, β取非负实数的时候，这个概率密度函数也都是良定义的。

好，我们回到魔鬼的游戏，这$n = 10, k = 7$这个具体的实例中，我们按照如下密度分布的峰值去猜测才是最有把握的。

$$f(x) = \frac{10!}{(6)!(3)!} x^6(1-x)^3 \quad x \in [0,1]$$

然而即便如此，我们能做到一次猜中的概率也不高，很不幸，你第一次没有猜中，魔鬼微笑着说：“我再仁慈一点，再给你一个机会，你按5下这个机器，你就得到了5个[0,1]之间的随机数，然后我可以告诉你这5个数中的每一个，和我的第7大的数相比，谁大谁小，然后你继续猜我手头的第7大的数是多少。”这时候我们应该怎么猜测呢？

0.3.2 Beta-Binomial 共轭

魔鬼的第二个题目，数学上形式化一下，就是

Algorithm 2 游戏2

1: $X_1, X_2, \cdots, X_n \overset{\text{iid}}{\sim} Uniform(0,1)$，排序后对应的顺序统计量$X_{(1)}, X_{(2)} \cdots, X_{(n)}$, 我们要猜测$p = X_{(k)}$；
2: $Y_1, Y_2, \cdots, Y_m \overset{\text{iid}}{\sim} Uniform(0,1)$, Y_i中有m_1个比p小，m_2个比p大；
3: 问$P(p|Y_1, Y_2, \cdots, Y_m)$ 的分布是什么。

由于$p = X_{(k)}$在$X_1, X_2, \cdots, X_n$中是第k大的，利用Y_i的信息，我们容易推理得到$p = X_{(k)}$ 在$X_1, X_2, \cdots, X_n, Y_1, Y_2, \cdots, Y_m \overset{\text{iid}}{\sim} Uniform(0,1)$ 这$(m+n)$个独立随机变量中是第$k+m_1$大的，于是按照上一个小节的推理，此时$p = X_{(k)}$的概率密度函数是$Beta(p|k+m_1, n-k+1+m_2)$。按照贝叶斯推理的逻辑，我们把以上过程整理如下：

1. $p = X_{(k)}$是我们要猜测的参数，我们推导出p 的分布为$f(p) = Beta(p|k, n-k+1)$,称为p 的先验分布；
2. 数据Y_i中有m_1个比p小，m_2个比p大，Y_i相当于是做了m次贝努利实验，所以m_1 服从二项分布$B(m,p)$；
3. 在给定了来自数据提供的(m_1, m_2)的知识后，p 的后验分布变为$f(p|m_1, m_2) = Beta(p|k+m_1, n-k+1+m_2)$

我们知道贝叶斯参数估计的基本过程是

先验分布+ 数据的知识= 后验分布

以上贝叶斯分析过程的简单直观的表述就是

$$Beta(p|k, n-k+1) + BinomCount(m_1, m_2) = Beta(p|k+m_1, n-k+1+m_2)$$

Figure 11: 贝努利实验

其中(m_1, m_2) 对应的是二项分布$B(m_1 + m_2, p)$的计数。更一般的，对于非负实数α, β，我们有如下关系

$$Beta(p|\alpha, \beta) + BinomCount(m_1, m_2) = Beta(p|\alpha + m_1, \beta + m_2) \tag{9}$$

这个式子实际上描述的就是**Beta-Binomial 共轭**，此处共轭的意思就是，数据符合二项分布的时候，参数的先验分布和后验分布都能保持Beta 分布的形式，这种形式不变的好处是，我们能够在先验分布中赋予参数很明确的物理意义，这个物理意义可以延续到后验分布中进行解释，同时从先验变换到后验过程中从数据中补充的知识也容易有物理解释。

而我们从以上过程可以看到，Beta 分布中的参数α, β都可以理解为物理计数，这两个参数经常被称为伪计数(pseudo-count)。基于以上逻辑，我们也可以把$Beta(p|\alpha, \beta)$写成下式来理解

$$Beta(p|1, 1) + BinomCount(\alpha - 1, \beta - 1) = Beta(p|\alpha, \beta) \tag{10}$$

其中$Beta(p|1, 1)$ 恰好就是均匀分布$Uniform(0, 1)$。

对于(10)式，我们其实也可以纯粹从贝叶斯的角度来进行推导和理解。假设有一个不均匀的硬币抛出正面的概率为p,抛m次后出现正面和反面的次数分别是m_1, m_2，那么按传统的频率学派观点，p的估计值应该为$\hat{p} = \frac{m_1}{m}$。而从贝叶斯学派的观点来看，开始对硬币不均匀性一无所知，所以应该假设$p \sim Uniform(0, 1)$, 于是有了二项分布的计数(m_1, m_2) 之后，按照贝叶斯公

式如下计算p 的后验分布

$$\begin{aligned} P(p|m_1, m_2) &= \frac{P(p) \cdot P(m_1, m_2|p)}{P(m_1, m_2)} \\ &= \frac{1 \cdot P(m_1, m_2|p)}{\int_0^1 P(m_1, m_2|t)dt} \\ &= \frac{\binom{m}{m_1} p^{m_1}(1-p)^{m_2}}{\int_0^1 \binom{m}{m_1} t^{m_1}(1-t)^{m_2} dt} \\ &= \frac{p^{m_1}(1-p)^{m_2}}{\int_0^1 t^{m_1}(1-t)^{m_2} dt} \end{aligned}$$

计算得到的后验分布正好是$Beta(p|m_1 + 1, m_2 + 1)$。

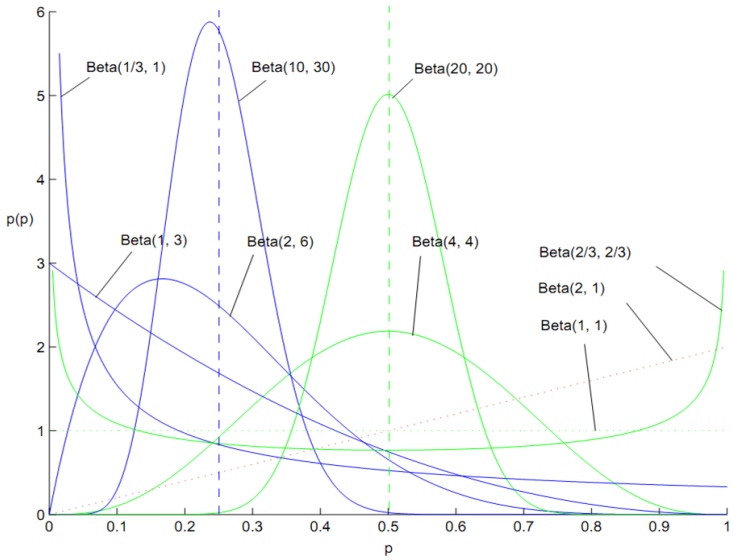


Figure 12: 百变星君Beta分布

Beta 分布的概率密度我们把它画成图，会发现它是个百变星君，它可以是凹的、凸的、单调上升的、单调下降的；可以是曲线也可以是直线，而均匀分布也是特殊的Beta分布，读者请参看一个网站上的Beta 分布Demo，通过调节参数α, β 可以观察Beta 分布的各种形态。。由于Beta 分布能够拟合如此之多的形状，因此它在统计数据拟合和贝叶斯分析中被广泛使用。

在上一个小节中，我们从二项分布推导Gamma 分布的时候，使用了如下

的等式

$$P(C \leq k) = \frac{n!}{k!(n-k-1)!} \int_p^1 t^k (1-t)^{n-k-1} dt, \quad C \sim B(n,p) \tag{11}$$

现在大家可以看到，左边是二项分布的概率累积，右边实际上是$Beta(t|k+1, n-k)$ 分布的概率积分。这个式子在上一小节中并没有给出证明，下面我们利用和魔鬼的游戏类似的概率物理过程进行证明。

我们可以如下构造二项分布，取随机变量$X_1, X_2, \cdots, X_n \overset{\text{iid}}{\sim} Uniform(0,1)$, 一个成功的贝努利实验就是$X_i < p$,否则表示失败,于是成功的概率为$p$。$C$用于计数成功的次数，于是$C \sim B(n,p)$。

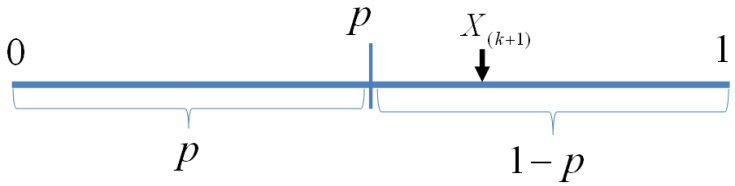


Figure 13: 贝努利实验最多成功k次

显然我们有如下式子成立

$$P(C \leq k) = P(X_{(k+1)} > p)$$

此处$X_{(k+1)}$是顺序统计量，为第$k+1$大的数。等式左边表示贝努利实验成功次数最多k次，右边表示第$k+1$ 大的数必然对应于失败的贝努利实验，从而失败次数最少是$n-k$次，所以左右两边是等价的。由于$X_{(k+1)} \sim Beta(t|k+1, n-k)$, 于是

$$\begin{aligned} P(C \leq k) &= P(X_{(k+1)} > p) \\ &= \int_p^1 Beta(t|k+1, n-k) dt \\ &= \frac{n!}{k!(n-k-1)!} \int_p^1 t^k (1-t)^{n-k-1} dt \end{aligned}$$

最后我们再回到魔鬼的游戏，如果你按出的5个随机数字中，魔鬼告诉你有2个小于它手中第7大的数，那么你应该按照如下概率分布的峰值做猜测是最好的

$$Beta(x|9,7) = \frac{15!}{(8)!(6)!} x^8 (1-x)^6 \quad x \in [0,1]$$

很幸运的，你这次猜中了，魔鬼开始甩赖了：这个游戏对你来说太简单了，我要加大点难度，我们重新来一次，我按魔盒20下生成20个随机数，你同时给我猜第7大和第13大的数是什么，这时候应该如何猜测呢？

0.3.3 Dirichlet-Multinomial 共轭

对于魔鬼变本加厉的新的游戏规则，数学形式化如下：

Algorithm 3 游戏3

1: $X_1, X_2, \cdots, X_n \overset{\text{iid}}{\sim} Uniform(0,1)$，
2: 排序后对应的顺序统计量$X_{(1)}, X_{(2)} \cdots, X_{(n)}$，
3: 问$(X_{(k_1)}, X_{(k_1+k_2)})$的联合分布是什么；

完全类似于第一个游戏的推导过程，我们可以进行如下的概率计算(为了数学公式的简洁对称，我们取x_3满足$x_1+x_2+x_3=1$,但只有x_1, x_2是变量)

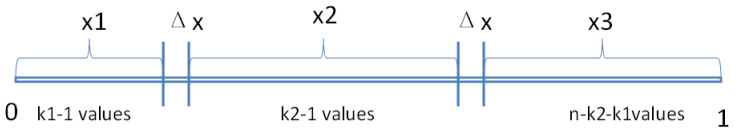


Figure 14: $(X_{(k_1)}, X_{(k_1+k_2)})$的联合分布推导

$$
\begin{aligned}
&P\Big(X_{(k_1)} \in (x_1, x_1+\Delta x), X_{(k_1+k_2)} \in (x_2, x_2+\Delta x)\Big) \\
&= n(n-1)\binom{n-2}{k_1-1, k_2-1} x_1^{k_1-1} x_2^{k_2-1} x_3^{n-k_1-k_2} (\Delta x)^2 \\
&= \frac{n!}{(k_1-1)!(k_2-1)!(n-k_1-k_2)!} x_1^{k_1-1} x_2^{k_2-1} x_3^{n-k_1-k_2} (\Delta x)^2
\end{aligned}
$$

于是我们得到$(X_{(k_1)}, X_{(k_1+k_2)})$的联合分布是

$$
\begin{aligned}
f(x_1, x_2, x_3) &= \frac{n!}{(k_1-1)!(k_2-1)!(n-k_1-k_2)!} x_1^{k_1-1} x_2^{k_2-1} x_3^{n-k_1-k_2} \\
&= \frac{\Gamma(n+1)}{\Gamma(k_1)\Gamma(k_2)\Gamma(n-k_1-k_2+1)} x_1^{k_1-1} x_2^{k_2-1} x_3^{n-k_1-k_2}
\end{aligned}
$$

熟悉Dirichlet的同学一眼就可以看出，上面这个分布其实就是3维形式的Dirichlet分布$Dir(x_1, x_2, x_3 | k_1, k_2, n-k_1-k_2+1)$。令$\alpha_1 = k_1, \alpha_2 = k_2, \alpha_3 = n-k_1-k_2+1$,于是分布密度可以写为

$$
f(x_1, x_2, x_3) = \frac{\Gamma(\alpha_1+\alpha_2+\alpha_3)}{\Gamma(\alpha_1)\Gamma(\alpha_2)\Gamma(\alpha_3)} x_1^{\alpha_1-1} x_2^{\alpha_2-1} x_3^{\alpha_3-1} \tag{12}
$$

这个就是一般形式的3维Dirichlet 分布，即便$\vec{\alpha}=(\alpha_1,\alpha_2,\alpha_3)$ 延拓到非负实数集合，以上概率分布也是良定义的。

从形式上我们也能看出，Dirichlet 分布是Beta 分布在高维度上的推广，他和Beta 分布一样也是一个百变星君，密度函数可以展现出多种形态。

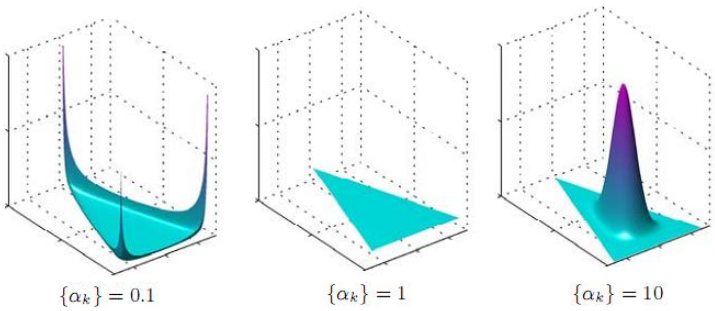


Figure 15: 不同α 下的Dirichlet 分布

类似于魔鬼的游戏2，我们也可以调整一下游戏3，从魔盒中生成m个随机数$Y_1,Y_2,\cdots,Y_m\overset{\text{iid}}{\sim}Uniform(0,1)$ 并让魔鬼告诉我们Y_i和$(X_{(k_1)},X_{(k_1+k_2)})$相比谁大谁小。于是有如下游戏4

Algorithm 4 游戏4

1: $X_1,X_2,\cdots,X_n\overset{\text{iid}}{\sim}Uniform(0,1)$，排序后对应的顺序统计量$X_{(1)},X_{(2)}\cdots,X_{(n)}$
2: 令$p_1=X_{(k_1)},p_2=X_{(k_1+k_2)},p_3=1-p_1-p_2$(加上$p_3$是为了数学表达简洁对称), 我们要猜测$\vec{p}=(p_1,p_2,p_3)$;
3: $Y_1,Y_2,\cdots,Y_m\overset{\text{iid}}{\sim}Uniform(0,1)$, Y_i中落到$[0,p_1),[p_1,p_2),[p_2,1]$ 三个区间的个数分别为m_1,m_2,m_3，$m=m_1+m_2+m3$;
4: 问后验分布$P(\vec{p}|Y_1,Y_2,\cdots,Y_m)$ 的分布是什么。

为了方便，我们记

$$\vec{m}=(m_1,m_2,m_3),\quad \vec{k}=(k_1,k_2,n-k_1-k_2+1)$$

由游戏中的信息，我们可以推理得到p_1,p_2在$X_1,X_2,\cdots,X_n,\ Y_1,Y_2,\cdots,Y_m\overset{\text{iid}}{\sim}Uniform(0,1)$ 这$m+n$个数中分别成为了第k_1+m_1,k_2+m_2大的数，于是后验分布$P(\vec{p}|Y_1,Y_2,\cdots,Y_m)$ 应该是$Dir(\vec{p}|k_1+m_1,k_1+m_2,n-k_1-k_2+1+m_3)$,即$Dir(\vec{p}|\vec{k}+\vec{m})$。按照贝叶斯推理的逻辑，我们同样可以把以上过程整理如下：

1. 我们要猜测参数$\vec{p}=(p_1,p_2,p_3)$，其先验分布为$Dir(\vec{p}|\vec{k})$；

2. 数据Y_i落到$[0,p_1),[p_1,p_2),[p_2,1]$三个区间的个数分别为$m_1,m_2,m_3$，所以$\vec{m}=(m_1,m_2,m_3)$ 服从多项分布$Mult(\vec{m}|\vec{p})$

3. 在给定了来自数据提供的知识$\vec{m}$后，$\vec{p}$ 的后验分布变为$Dir(\vec{p}|\vec{k}+\vec{m})$

以上贝叶斯分析过程的简单直观的表述就是

$$Dir(\vec{p}|\vec{k})+MultCount(\vec{m})=Dir(\vec{p}|\vec{k}+\vec{m})$$

令$\vec{\alpha}=\vec{k}$,把$\vec{\alpha}$ 从整数集合延拓到实数集合，更一般的可以证明有如下关系

$$Dir(\vec{p}|\vec{\alpha})+MultCount(\vec{m})=Dir(\vec{p}|\vec{\alpha}+\vec{m}) \tag{13}$$

以上式子实际上描述的就是**Dirichlet-Multinomial 共轭**，而我们从以上过程可以看到，Dirichlet 分布中的参数$\vec{\alpha}$都可以理解为物理计数。类似于Beta 分布，我们也可以把$Dir(\vec{p}|\vec{\alpha})$作如下分解

$$Dir(\vec{p}|\vec{1})+MultCount(\vec{m}-\vec{1})=Dir(\vec{p}|\vec{\alpha})$$

此处$\vec{1}=(1,1,\cdots,1)$。自然，上式我们也可以类似地用纯粹贝叶斯的观点进行推导和解释。

以上的游戏我们还可以往更高的维度上继续推，譬如猜测$X_{(1)},X_{(2)}\cdots,X_{(n)}$中的4、5、...等更多个数，于是就得到更高纬度的Dirichlet 分布和Dirichlet-Multinomial 共轭。一般形式的Dirichlet 分布定义如下

$$Dir(\vec{p}|\vec{\alpha})=\frac{\Gamma(\sum_{k=1}^{K}\alpha_k)}{\prod_{k=1}^{K}\Gamma(\alpha_k)}\prod_{k=1}^{K}p_k^{\alpha_k-1} \tag{14}$$

对于给定的$\vec{p}$和N,多项分布定义为

$$Mult(\vec{n}|\vec{p},N)=\binom{N}{\vec{n}}\prod_{k=1}^{K}p_k^{n_k} \tag{15}$$

而$Mult(\vec{n}|\vec{p},N)$ 和$Dir(\vec{p}|\vec{\alpha})$ 这两个分布是共轭关系。

Beta-Binomail 共轭和Dirichlet-Multinomail 共轭都可以用纯粹数学的方式进行证明，我们在这两个小节中通过一个游戏来解释这两个共轭关系，主要是想说明这个共轭关系是可以对应到很具体的概率物理过程的。

0.3.4 Beta/Dirichlet 分布的一个性质

如果$p \sim Beta(t|\alpha,\beta)$, 则

$$\begin{aligned} E(p) &= \int_0^1 t * Beta(t|\alpha,\beta)dt \\ &= \int_0^1 t * \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} t^{\alpha-1}(1-t)^{\beta-1}dt \\ &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \int_0^1 t^{\alpha}(1-t)^{\beta-1}dt \end{aligned}$$

上式右边的积分对应到概率分布$Beta(t|\alpha+1,\beta)$，对于这个分布，我们有

$$\int_0^1 \frac{\Gamma(\alpha+\beta+1)}{\Gamma(\alpha+1)\Gamma(\beta)} t^{\alpha}(1-t)^{\beta-1}dt = 1$$

把上式带入$E(p)$的计算式，得到

$$\begin{aligned} E(p) &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} \cdot \frac{\Gamma(\alpha+1)\Gamma(\beta)}{\Gamma(\alpha+\beta+1)} \\ &= \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha+\beta+1)} \frac{\Gamma(\alpha+1)}{\Gamma(\alpha)} \\ &= \frac{\alpha}{\alpha+\beta} \end{aligned} \tag{16}$$

这说明，对于Beta 分布的随机变量，其均值可以用$\frac{\alpha}{\alpha+\beta}$来估计。Dirichlet 分布也有类似的结论，如果$\vec{p} \sim Dir(\vec{t}|\vec{\alpha})$，同样可以证明

$$E(\vec{p}) = \Big(\frac{\alpha_1}{\sum_{i=1}^K \alpha_i}, \frac{\alpha_2}{\sum_{i=1}^K \alpha_i}, \cdots, \frac{\alpha_K}{\sum_{i=1}^K \alpha_i}\Big) \tag{17}$$

以上两个结论很重要，因为我们在后面的LDA 数学推导中需要使用这个结论。

0.4 MCMC 和Gibbs Sampling

0.4.1 随机模拟

随机模拟(或者统计模拟)方法有一个很酷的别名是蒙特卡罗方法(Monte Carlo Simulation)。这个方法的发展始于20世纪40年代，和原子弹制造的曼哈顿计划密切相关，当时的几个大牛，包括乌拉姆、冯.诺依曼、费米、费曼、Nicholas Metropolis，在美国洛斯阿拉莫斯国家实验室研究裂变物质的中子连锁反应的时候，开始使用统计模拟的方法,并在最早的计算机上进行编程实现。

现代的统计模拟方法最早由数学家乌拉姆提出，被Metropolis命名为蒙特卡罗方法，蒙特卡罗是著名的赌场，赌博总是和统计密切关联的，所以这个命

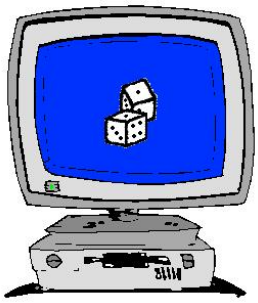


Figure 16: 随机模拟与计算机

名风趣而贴切，很快被大家广泛接受。被不过据说费米之前就已经在实验中使用了，但是没有发表。说起蒙特卡罗方法的源头，可以追溯到18世纪，布丰当年用于计算π的著名的投针实验就是蒙特卡罗模拟实验。统计采样的方法其实数学家们很早就知道，但是在计算机出现以前，随机数生成的成本很高，所以该方法也没有实用价值。随着计算机技术在二十世纪后半叶的迅猛发展，随机模拟技术很快进入实用阶段。对那些用确定算法不可行或不可能解决的问题，蒙特卡罗方法常常为人们带来希望。

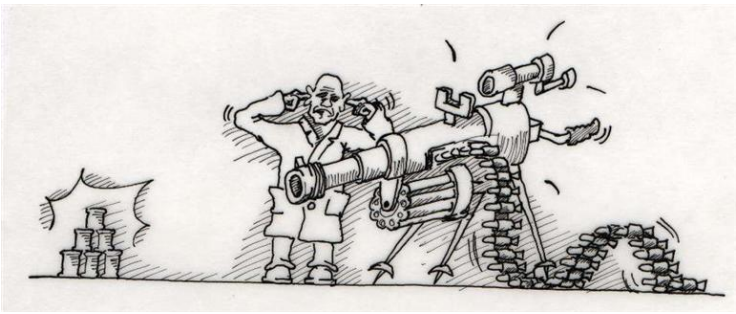


Figure 17: 蒙特卡罗方法

统计模拟中有一个重要的问题就是给定一个概率分布$p(x)$，我们如何在计算机中生成它的样本。一般而言均匀分布$Uniform(0,1)$的样本是相对容易生成的。通过线性同余发生器可以生成伪随机数，我们用确定性算法生成$[0,1]$之间的伪随机数序列后，这些序列的各种统计指标和均匀分布$Uniform(0,1)$ 的理论计算结果非常接近。这样的伪随机序列就有比较好的统计性质，可以被当成真实的随机数使用。

而我们常见的概率分布，无论是连续的还是离散的分布，都可以基

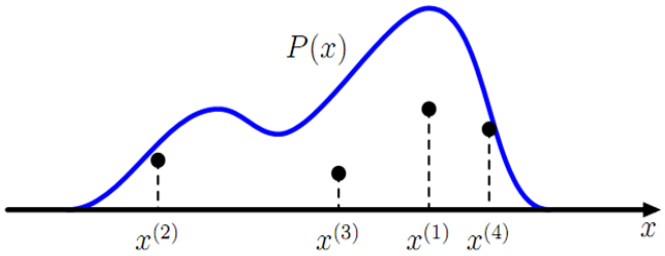


Figure 18: 生成一个概率分布的样本

于$Uniform(0,1)$ 的样本生成。例如正态分布可以通过著名的Box-Muller 变换得到

定理0.4.1 (Box-Muller **变换**)　如果随机变量U_1, U_2 独立且$U_1, U_2 \sim Uniform[0,1]$，则

$$Z_0 = \sqrt{-2\ln U_1} cos(2\pi U_2)$$
$$Z_1 = \sqrt{-2\ln U_1} sin(2\pi U_2)$$

则，Z_0, Z_1 独立且服从标准正态分布。

其它几个著名的连续分布，包括指数分布、Gamma 分布、t 分布、F 分布、Beta 分布、Dirichlet 分布等等,也都可以通过类似的数学变换得到；离散的分布通过均匀分布更加容易生成。更多的统计分布如何通过均匀分布的变换生成出来，大家可以参考统计计算的书，其中Sheldon M. Ross 的《统计模拟》是写得非常通俗易懂的一本。

不过我们并不是总是这么幸运的，当$p(x)$的形式很复杂，或者$p(\mathbf{x})$ 是个高维的分布的时候，样本的生成就可能很困难了。譬如有如下的情况

1. $p(x) = \dfrac{\tilde{p}(x)}{\int \tilde{p}(x)dx}$, 而$\tilde{p}(x)$ 我们是可以计算的，但是底下的积分式无法显式计算。

2. $p(x,y)$ 是一个二维的分布函数，这个函数本身计算很困难，但是条件分布$p(x|y), p(y|x)$的计算相对简单; 如果$p(\mathbf{x})$ 是高维的，这种情形就更加明显。

此时就需要使用一些更加复杂的随机模拟的方法来生成样本。而本节中将要重点介绍的MCMC(Markov Chain Monte Carlo) 和Gibbs Sampling算法就是

最常用的一种，这两个方法在现代贝叶斯分析中被广泛使用。要了解这两个算法，我们首先要对马氏链的平稳分布的性质有基本的认识。

0.4.2 马氏链及其平稳分布

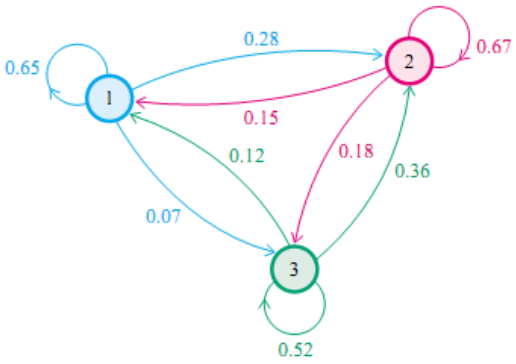
马氏链的数学定义很简单

$$P(X_{t+1} = x|X_t, X_{t-1}, \cdots) = P(X_{t+1} = x|X_t)$$

也就是状态转移的概率只依赖于前一个状态。

我们先来看马氏链的一个具体的例子。社会学家经常把人按其经济状况分成3类：下层(lower-class)、中层(middle-class)、上层(upper-class)，我们用1,2,3分别代表这三个阶层。社会学家们发现决定一个人的收入阶层的最重要的因素就是其父母的收入阶层。如果一个人的收入属于下层类别，那么他的孩子属于下层收入的概率是0.65, 属于中层收入的概率是0.28, 属于上层收入的概率是0.07。事实上，从父代到子代，收入阶层的变化的转移概率如下

		子代		
	State	1	2	3
父代	1	0.65	0.28	0.07
	2	0.15	0.67	0.18
	3	0.12	0.36	0.52



使用矩阵的表示方式，转移概率矩阵记为

$$P = \begin{bmatrix} 0.65 & 0.28 & 0.07 \\ 0.15 & 0.67 & 0.18 \\ 0.12 & 0.36 & 0.52 \end{bmatrix}$$

假设当前这一代人处在下层、中层、上层的人的比例是概率分布向量$\pi_0 = [\pi_0(1), \pi_0(2), \pi_0(3)]$，那么他们的子女的分布比例将是$\pi_1 = \pi_0 P$, 他们的孙子代的分布比例将是$\pi_2 = \pi_1 P = \pi_0 P^2$, ..., 第$n$代子孙的收入分布比例将是$\pi_n = \pi_{n-1} P = \pi_0 P^n$。

假设初始概率分布为$\pi_0 = [0.21, 0.68, 0.11]$，则我们可以计算前$n$代人的分布状况如下

第n代人	下层	中层	上层
0	0.210	0.680	0.110
1	0.252	0.554	0.194
2	0.270	0.512	0.218
3	0.278	0.497	0.225
4	0.282	0.490	0.226
5	0.285	0.489	0.225
6	0.286	0.489	0.225
7	0.286	0.489	0.225
8	0.289	0.488	0.225
9	0.286	0.489	0.225
10	0.286	0.489	0.225
...	...	...	...

我们发现从第7代人开始，这个分布就稳定不变了，这个是偶然的吗？我们换一个初始概率分布$\pi_0 = [0.75, 0.15, 0.1]$ 试试看，继续计算前n代人的分布状况如下

第n代人	下层	中层	上层
0	0.75	0.15	0.1
1	0.522	0.347	0.132
2	0.407	0.426	0.167
3	0.349	0.459	0.192
4	0.318	0.475	0.207
5	0.303	0.482	0.215
6	0.295	0.485	0.220
7	0.291	0.487	0.222
8	0.289	0.488	0.225
9	0.286	0.489	0.225
10	0.286	0.489	0.225
...	...	...	...

我们发现，到第9代人的时候, 分布又收敛了。最为奇特的是，两次给定不同的初始概率分布，最终都收敛到概率分布$\pi = [0.286, 0.489, 0.225]$，也就是说收敛的行为和初始概率分布$\pi_0$ 无关。这说明这个收敛行为主要是由概率转移矩阵P决定的。我们计算一下P^n

$$P^{20} = P^{21} = \cdots = P^{100} = \cdots = \begin{bmatrix} 0.286 & 0.489 & 0.225 \\ 0.286 & 0.489 & 0.225 \\ 0.286 & 0.489 & 0.225 \end{bmatrix}$$

我们发现，当n 足够大的时候，这个P^n矩阵的每一行都是稳定地收敛到$\pi = [0.286, 0.489, 0.225]$ 这个概率分布。自然的，这个收敛现象并非是我们这个马氏链独有的，而是绝大多数马氏链的共同行为，关于马氏链的收敛我们有如下漂亮的定理：

定理0.4.2　如果一个非周期马氏链具有转移概率矩阵P,且它的任何两个状态是连通的，那么$\lim\limits_{n\to\infty} P_{ij}^n$ 存在且与i无关，记$\lim\limits_{n\to\infty} P_{ij}^n = \pi(j)$, 我们有

1. $\lim\limits_{n\to\infty} P^n = \begin{bmatrix} \pi(1) & \pi(2) & \cdots & \pi(j) & \cdots \\ \pi(1) & \pi(2) & \cdots & \pi(j) & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \pi(1) & \pi(2) & \cdots & \pi(j) & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix}$
2. $\pi(j) = \sum\limits_{i=0}^{\infty} \pi(i) P_{ij}$
3. π 是方程$\pi P = \pi$ 的唯一非负解

其中,

$$\pi = [\pi(1), \pi(2), \cdots, \pi(j), \cdots], \quad \sum_{i=0}^{\infty} \pi_i = 1$$

π称为马氏链的平稳分布。

这个马氏链的收敛定理非常重要，**所有的MCMC(Markov Chain Monte Carlo) 方法都是以这个定理作为理论基础的**。 定理的证明相对复杂，一般的随机过程课本中也不给证明，所以我们就不用纠结它的证明了，直接用这个定理的结论就好了。我们对这个定理的内容做一些解释说明：

1. 该定理中马氏链的状态不要求有限，可以是有无穷多个的；
2. 定理中的“非周期“这个概念我们不打算解释了，因为我们遇到的绝大多数马氏链都是非周期的；

3. 两个状态i, j是连通并非指i 可以直接一步转移到$j(P_{ij} > 0)$,而是指i 可以通过有限的n步转移到达$j(P_{ij}^n > 0)$。马氏链的任何两个状态是连通的含义是指存在一个n, 使得矩阵P^n 中的任何一个元素的数值都大于零。

4. 我们用X_i 表示在马氏链上跳转第i步后所处的状态，如果$\lim_{n\to\infty} P_{ij}^n = \pi(j)$存在，很容易证明以上定理的第二个结论。由于

$$\begin{aligned} P(X_{n+1} = j) &= \sum_{i=0}^{\infty} P(X_n = i)P(X_{n+1} = j|X_n = i) \\ &= \sum_{i=0}^{\infty} P(X_n = i)P_{ij} \end{aligned}$$

上式两边取极限就得到$\pi(j) = \sum_{i=0}^{\infty} \pi(i)P_{ij}$

从初始概率分布π_0 出发，我们在马氏链上做状态转移，记X_i的概率分布为π_i, 则有

$$\begin{aligned} &X_0 \sim \pi_0(x) \\ &X_i \sim \pi_i(x), \qquad \pi_i(x) = \pi_{i-1}(x)P = \pi_0(x)P^n \end{aligned}$$

由马氏链收敛的定理, 概率分布$\pi_i(x)$将收敛到平稳分布$\pi(x)$。假设到第n步的时候马氏链收敛，则有

$$\begin{aligned} X_0 &\sim \pi_0(x) \\ X_1 &\sim \pi_1(x) \\ &\cdots \\ X_n &\sim \pi_n(x) = \pi(x) \\ X_{n+1} &\sim \pi(x) \\ X_{n+2} &\sim \pi(x) \\ &\cdots \end{aligned}$$

所以$X_n, X_{n+1}, X_{n+2}, \cdots \sim \pi(x)$ 都是同分布的随机变量，当然他们并不独立。如果我们从一个具体的初始状态x_0 开始,沿着马氏链按照概率转移矩阵做跳转，那么我们得到一个转移序列$x_0, x_1, x_2, \cdots x_n, x_{n+1} \cdots$, 由于马氏链的收敛行为，$x_n, x_{n+1}, \cdots$ 都将是平稳分布$\pi(x)$ 的样本。

0.4.3 Markov Chain Monte Carlo

对于给定的概率分布$p(x)$,我们希望能有便捷的方式生成它对应的样本。由于马氏链能收敛到平稳分布，于是一个很的漂亮想法是：如果我们能构造一个转移矩阵为P的马氏链，使得该马氏链的平稳分布恰好是$p(x)$, 那么我们从任何一个初始状态x_0出发沿着马氏链转移, 得到一个转移序列$x_0, x_1, x_2, \cdots x_n, x_{n+1} \cdots$,，如果马氏链在第$n$步已经收敛了，于是我们就得到了$\pi(x)$ 的样本$x_n, x_{n+1} \cdots$。

这个绝妙的想法在1953年被Metropolis想到了，为了研究粒子系统的平稳性质，Metropolis 考虑了物理学中常见的波尔兹曼分布的采样问题，首次提出了基于马氏链的蒙特卡罗方法，即Metropolis算法，并在最早的计算机上编程实现。Metropolis 算法是首个普适的采样方法，并启发了一系列MCMC方法，所以人们把它视为随机模拟技术腾飞的起点。Metropolis的这篇论文被收录在《统计学中的重大突破》中，Metropolis算法也被遴选为二十世纪的十个最重要的算法之一。

我们接下来介绍的MCMC 算法是Metropolis 算法的一个改进变种，即常用的Metropolis-Hastings 算法。由上一节的例子和定理我们看到了，马氏链的收敛性质主要由转移矩阵P 决定, 所以基于马氏链做采样的关键问题是如何构造转移矩阵P,使得平稳分布恰好是我们要的分布$p(x)$。如何能做到这一点呢？我们主要使用如下的定理。

定理0.4.3 (**细致平稳条件**) 如果非周期马氏链的转移矩阵P和分布$\pi(x)$ 满足

$$\pi(i)P_{ij} = \pi(j)P_{ji} \qquad \text{for all} \quad i, j \tag{18}$$

则$\pi(x)$ 是马氏链的平稳分布，上式被称为细致平稳条件(detailed balance condition)。

其实这个定理是显而易见的，因为细致平稳条件的物理含义就是对于任何两个状态i, j, 从i 转移出去到j 而丢失的概率质量，恰好会被从j 转移回i 的概率质量补充回来，所以状态i上的概率质量$\pi(i)$是稳定的，从而$\pi(x)$是马氏链的平稳分布。数学上的证明也很简单，由细致平稳条件可得

$$\sum_{i=1}^{\infty} \pi(i)P_{ij} = \sum_{i=1}^{\infty} \pi(j)P_{ji} = \pi(j) \sum_{i=1}^{\infty} P_{ji} = \pi(j)$$
$$\Rightarrow \pi P = \pi$$

由于π 是方程$\pi P = \pi$的解，所以π是平稳分布。

假设我们已经有一个转移矩阵为Q马氏链($q(i,j)$表示从状态i转移到状态j的概率，也可以写为$q(j|i)$或者$q(i \to j)$), 显然，通常情况下

$$p(i)q(i,j) \neq p(j)q(j,i)$$

也就是细致平稳条件不成立，所以$p(x)$ 不太可能是这个马氏链的平稳分布。我们可否对马氏链做一个改造，使得细致平稳条件成立呢？譬如，我们引入一个$\alpha(i,j)$, 我们希望

$$p(i)q(i,j)\alpha(i,j) = p(j)q(j,i)\alpha(j,i) \tag{19}$$

取什么样的$\alpha(i,j)$ 以上等式能成立呢？最简单的，按照对称性，我们可以取

$$\alpha(i,j) = p(j)q(j,i) \quad \alpha(j,i) = p(i)q(i,j)$$

于是(19)式就成立了。所以有

$$p(i)\underbrace{q(i,j)\alpha(i,j)}_{Q'(i,j)} = p(j)\underbrace{q(j,i)\alpha(j,i)}_{Q'(j,i)} \tag{20}$$

于是我们把原来具有转移矩阵Q的一个很普通的马氏链，改造为了具有转移矩阵Q'的马氏链，而Q'恰好满足细致平稳条件，由此马氏链Q'的平稳分布就是$p(x)$!

在改造Q 的过程中引入的$\alpha(i,j)$称为接受率，物理意义可以理解为在原来的马氏链上，从状态i 以$q(i,j)$ 的概率转跳转到状态j 的时候，我们以$\alpha(i,j)$的概率接受这个转移，于是得到新的马氏链Q'的转移概率为$q(i,j)\alpha(i,j)$。

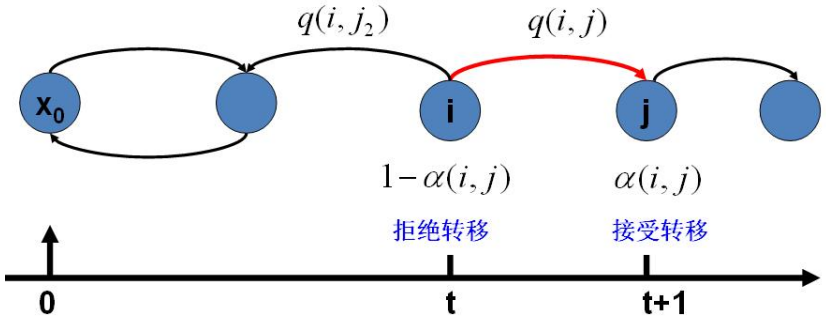


Figure 19: 马氏链转移和接受概率

假设我们已经有一个转移矩阵Q(对应元素为$q(i,j)$), 把以上的过程整理一下，我们就得到了如下的用于采样概率分布$p(x)$的算法。

Algorithm 5 MCMC 采样算法

1: 初始化马氏链初始状态$X_0 = x_0$

2: 对$t = 0, 1, 2, \cdots$, 循环以下过程进行采样

- 第t个时刻马氏链状态为$X_t = x_t$, 采样$y \sim q(x|x_t)$
- 从均匀分布采样$u \sim Uniform[0, 1]$
- 如果$u < \alpha(x_t, y) = p(y)q(x_t|y)$ 则接受转移$x_t \to y$，即$X_{t+1} = y$
- 否则不接受转移，即$X_{t+1} = x_t$

上述过程中$p(x), q(x|y)$ 说的都是离散的情形，事实上即便这两个分布是连续的，以上算法仍然是有效，于是就得到更一般的连续概率分布$p(x)$的采样算法，而$q(x|y)$ 就是任意一个连续二元概率分布对应的条件分布。

以上的MCMC 采样算法已经能很漂亮的工作了，不过它有一个小的问题：马氏链Q在转移的过程中的接受率$\alpha(i, j)$ 可能偏小，这样采样过程中马氏链容易原地踏步，拒绝大量的跳转，这使得马氏链遍历所有的状态空间要花费太长的时间，收敛到平稳分布$p(x)$的速度太慢。有没有办法提升一些接受率呢?

假设$\alpha(i, j) = 0.1, \alpha(j, i) = 0.2$, 此时满足细致平稳条件，于是

$$p(i)q(i, j) \times 0.1 = p(j)q(j, i) \times 0.2$$

上式两边扩大5倍，我们改写为

$$p(i)q(i, j) \times 0.5 = p(j)q(j, i) \times 1$$

看，我们提高了接受率，而细致平稳条件并没有打破！这启发我们可以把细致平稳条件(20)式中的$\alpha(i, j), \alpha(j, i)$ 同比例放大，使得两数中最大的一个放大到1，这样我们就提高了采样中的跳转接受率。所以我们可以取

$$\alpha(i, j) = \min\left\{\frac{p(j)q(j, i)}{p(i)q(i, j)}, 1\right\}$$

于是，经过对上述MCMC 采样算法中接受率的微小改造，我们就得到了如下教科书中最常见的Metropolis-Hastings 算法。

对于分布$p(x)$,我们构造转移矩阵Q' 使其满足细致平稳条件

$$p(x)Q'(x \to y) = p(y)Q'(y \to x)$$

此处x 并不要求是一维的，对于高维空间的$p(\mathbf{x})$，如果满足细致平稳条件

$$p(\mathbf{x})Q'(\mathbf{x} \to \mathbf{y}) = p(\mathbf{y})Q'(\mathbf{y} \to \mathbf{x})$$

Algorithm 6 Metropolis-Hastings 采样算法

1: 初始化马氏链初始状态$X_0 = x_0$

2: 对$t = 0, 1, 2, \cdots$, 循环以下过程进行采样

- 第t个时刻马氏链状态为$X_t = x_t$, 采样$y \sim q(x|x_t)$
- 从均匀分布采样$u \sim Uniform[0, 1]$
- 如果$u < \alpha(x_t, y) = \min\left\{\frac{p(y)q(x_t|y)}{p(x_t)p(y|x_t)}, 1\right\}$则接受转移$x_t \to y$，即$X_{t+1} = y$
- 否则不接受转移，即$X_{t+1} = x_t$

那么以上的Metropolis-Hastings 算法一样有效。

0.4.4 Gibbs Sampling

对于高维的情形，由于接受率α的存在(通常$\alpha < 1$), 以上Metropolis-Hastings 算法的效率不够高。能否找到一个转移矩阵Q使得接受率$\alpha = 1$呢？我们先看看二维的情形，假设有一个概率分布$p(x, y)$, 考察x坐标相同的两个点$A(x_1, y_1), B(x_1, y_2)$，我们发现

$$p(x_1, y_1)p(y_2|x_1) = p(x_1)p(y_1|x_1)p(y_2|x_1)$$
$$p(x_1, y_2)p(y_1|x_1) = p(x_1)p(y_2|x_1)p(y_1|x_1)$$

所以得到

$$p(x_1, y_1)p(y_2|x_1) = p(x_1, y_2)p(y_1|x_1) \tag{21}$$

即

$$p(A)p(y_2|x_1) = p(B)p(y_1|x_1)$$

基于以上等式，我们发现，在$x = x_1$ 这条平行于y轴的直线上，如果使用条件分布$p(y|x_1)$做为任何两个点之间的转移概率，那么任何两个点之间的转移满足细致平稳条件。同样的，如果我们在$y = y_1$ 这条直线上任意取两个点$A(x_1, y_1), C(x_2, y_1)$,也有如下等式

$$p(A)p(x_2|y_1) = p(C)p(x_1|y_1).$$

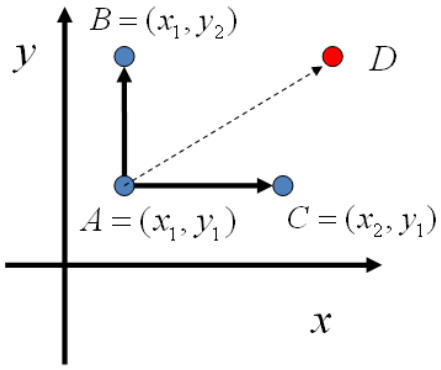


Figure 20: 平面上马氏链转移矩阵的构造

于是我们可以如下构造平面上任意两点之间的转移概率矩阵Q

$$\begin{aligned} Q(A \to B) &= p(y_B|x_1) && \text{如果} \quad x_A = x_B = x_1 \\ Q(A \to C) &= p(x_C|y_1) && \text{如果} \quad y_A = y_C = y_1 \\ Q(A \to D) &= 0 && \text{其它} \end{aligned}$$

有了如上的转移矩阵Q, 我们很容易验证对平面上任意两点X, Y, 满足细致平稳条件

$$p(X)Q(X \to Y) = p(Y)Q(Y \to X)$$

于是这个二维空间上的马氏链将收敛到平稳分布$p(x, y)$。而这个算法就称为Gibbs Sampling 算法,由物理学家Gibbs 首先给出的。

Algorithm 7 二维Gibbs Sampling 算法

1: 随机初始化$X_0 = x_0 Y_0 = y_0$

2: 对$t = 0, 1, 2, \cdots$ 循环采样

1. $y_{t+1} \sim p(y|x_t)$
2. $x_{t+1} \sim p(x|y_{t+1})$

以上采样过程中，如图所示，马氏链的转移只是轮换的沿着坐标轴x轴和y轴做转移，于是得到样本$(x_0, y_0), (x_0, y_1), (x_1, y_1), (x_1, y_2), (x_2, y_2), \cdots$ 马氏链收敛后，最终得到的样本就是$p(x, y)$ 的样本，而收敛之前的阶段称为burn-in period。额外说明一下，我们看到教科书上的Gibbs Sampling 算法大都是坐标

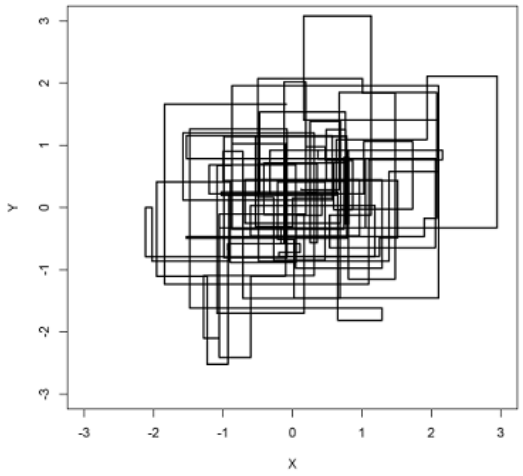


Figure 21: 二维Gibbs Sampling 算法中的马氏链转移

轴轮换采样的，但是这其实是不强制要求的。最一般的情形可以是，在t时刻，可以在x轴和y轴之间随机的选一个坐标轴，然后按条件概率做转移，马氏链也是一样收敛的。轮换两个坐标轴只是一种方便的形式。

以上的过程我们很容易推广到高维的情形，对于(21)式，如果x_1 变为多维情形$\mathbf{x_1}$，可以看出推导过程不变，所以细致平稳条件同样是成立的

$$p(\mathbf{x_1}, y_1)p(y_2|\mathbf{x_1}) = p(\mathbf{x_1}, y_2)p(y_1|\mathbf{x_1}) \tag{22}$$

此时转移矩阵Q 由条件分布$p(y|\mathbf{x_1})$ 定义。上式只是说明了一根坐标轴的情形，和二维情形类似，很容易验证对所有坐标轴都有类似的结论。所以n维空间中对于概率分布$p(x_1, x_2, \cdots, x_n)$ 可以如下定义转移矩阵

1. 如果当前状态为$(x_1, x_2, \cdots, x_n)$，马氏链转移的过程中，只能沿着坐标轴做转移。沿着x_i 这根坐标轴做转移的时候，转移概率由条件概率$p(x_i|x_1, \cdots, x_{i-1}, x_{i+1}, \cdots, x_n)$ 定义；

2. 其它无法沿着单根坐标轴进行的跳转，转移概率都设置为0。

于是我们可以把Gibbs Smapling 算法从采样二维的$p(x, y)$ 推广到采样n 维的$p(x_1, x_2, \cdots, x_n)$

以上算法收敛后，得到的就是概率分布$p(x_1, x_2, \cdots, x_n)$的样本，当然这些样本并不独立，但是我们此处要求的是采样得到的样本符合给定的概率分布，并不要求独立。同样的，在以上算法中，坐标轴轮换采样不是必须的，可以在

Algorithm 8 n维Gibbs Sampling 算法

1: 随机初始化$\{x_i : i = 1, \cdots, n\}$

2: 对$t = 0, 1, 2, \cdots$ 循环采样

1. $x_1^{(t+1)} \sim p(x_1|x_2^{(t)}, x_3^{(t)}, \cdots, x_n^{(t)})$
2. $x_2^{(t+1)} \sim p(x_2|x_1^{(t+1)}, x_3^{(t)}, \cdots, x_n^{(t)})$
3. $\cdots$
4. $x_j^{(t+1)} \sim p(x_j|x_1^{(t+1)}, \cdots, x_{j-1}^{(t+1)}, x_{j+1}^{(t)}, \cdots, x_n^{(t)})$
5. $\cdots$
6. $x_n^{(t+1)} \sim p(x_n|x_1^{(t+1)}, x_2^t, \cdots, x_{n-1}^{(t+1)})$

坐标轴轮换中引入随机性，这时候转移矩阵Q 中任何两个点的转移概率中就会包含坐标轴选择的概率，而在通常的Gibbs Sampling 算法中，坐标轴轮换是一个确定性的过程，也就是在给定时刻t，在一根固定的坐标轴上转移的概率是1。

0.5 文本建模

我们日常生活中总是产生大量的文本，如果每一个文本存储为一篇文档，那每篇文档从人的观察来说就是有序的词的序列$d = (w_1, w_2, \cdots, w_n)$。

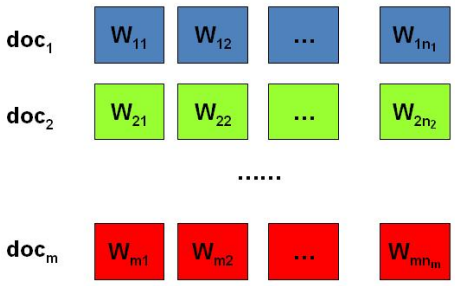


Figure 22: 包含m 篇文档的语料库

统计文本建模的目的就是追问这些观察到语料库中的的词序列是如何生成的。统计学被人们描述为猜测上帝的游戏，人类产生的所有的语料文本我们都

可以看成是一个伟大的上帝在天堂中抛掷骰子生成的，我们观察到的只是上帝玩这个游戏的结果—— 词序列构成的语料，而上帝玩这个游戏的过程对我们是个黑盒子。所以在统计文本建模中，我们希望猜测出上帝是如何玩这个游戏的，具体一点，最核心的两个问题是

1. 上帝都有什么样的骰子；
2. 上帝是如何抛掷这些骰子的；

第一个问题就是表示模型中都有哪些参数，骰子的每一个面的概率都对应于模型中的参数；第二个问题就表示游戏规则是什么，上帝可能有各种不同类型的骰子，上帝可以按照一定的规则抛掷这些骰子从而产生词序列。

Figure 23: 上帝掷骰子

0.5.1 Unigram Model

假设我们的词典中一共有V 个词$v_1, v_2, \cdots v_V$，那么最简单的Unigram Model 就是认为上帝是按照如下的游戏规则产生文本的。

Game 9 Unigram Model

1: 上帝只有一个骰子，这个骰子有V 个面，每个面对应一个词, 各个面的概率不一；
2: 每抛一次骰子，抛出的面就对应的产生一个词；如果一篇文档中有n 个词，上帝就是独立的抛n次骰子产生这n 个词；

上帝的这个唯一的骰子各个面的概率记为$\vec{p} = (p_1, p_2, \cdots, p_V)$, 所以每次投掷骰子类似于一个抛钢镚时候的贝努利实验，只是贝努利实验中我们抛的是一个两面的骰子，而此处抛的是一个V面的骰子，我们把抛这个V面骰子的实验记为记为$w \sim Mult(w|\vec{p})$。

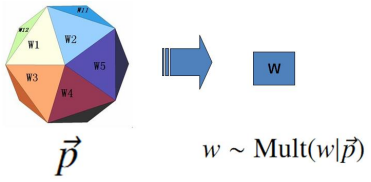


Figure 24: 上帝投掷V 个面的骰子

对于一篇文档$d = \vec{w} = (w_1, w_2, \cdots, w_n)$, 该文档被生成的概率就是

$$p(\vec{w}) = p(w_1, w_2, \cdots, w_n) = p(w_1)p(w_2)\cdots p(w_n)$$

而文档和文档之间我们认为是独立的，所以如果语料中有多篇文档$\mathcal{W} = (\vec{w_1}, \vec{w_2}, ..., \vec{w_m})$, 则该语料的概率是

$$p(\mathcal{W}) = p(\vec{w_1})p(\vec{w_2})\cdots p(\vec{w_m})$$

在Unigram Model 中，我们假设了文档之间是独立可交换的，而文档中的词也是独立可交换的，所以一篇文档相当于一个袋子，里面装了一些词，而词的顺序信息就无关紧要了，这样的模型也称为词袋模型(Bag-of-words)。

假设语料中总的词频是N, 在所有的N 个词中,如果我们关注每个词v_i 的发生次数n_i，那么$\vec{n} = (n_1, n_2, \cdots, n_V)$ 正好是一个多项分布

$$p(\vec{n}) = Mult(\vec{n}|\vec{p}, N) = \binom{N}{\vec{n}} \prod_{k=1}^{V} p_k^{n_k}$$

此时，语料的概率是

$$p(\mathcal{W}) = p(\vec{w_1})p(\vec{w_2})\cdots p(\vec{w_m}) = \prod_{k=1}^{V} p_k^{n_k}$$

当然，我们很重要的一个任务就是估计模型中的参数$\vec{p}$，也就是问上帝拥有的这个骰子的各个面的概率是多大，按照统计学家中频率派的观点，使用最大似然估计最大化$P(\mathcal{W})$，于是参数p_i的估计值就是

$$\hat{p_i} = \frac{n_i}{N}.$$

对于以上模型，贝叶斯统计学派的统计学家会有不同意见，他们会很挑剔的批评只假设上帝拥有唯一一个固定的骰子是不合理的。在贝叶斯学派看来，一切参数都是随机变量，以上模型中的骰子$\vec{p}$不是唯一固定的，它也是一个随机变量。所以按照贝叶斯学派的观点，上帝是按照以下的过程在玩游戏的

Game 10 贝叶斯Unigram Model假设

1: 上帝有一个装有无穷多个骰子的坛子，里面有各式各样的骰子，每个骰子有V 个面；

2: 上帝从坛子里面抽了一个骰子出来，然后用这个骰子不断的抛，然后产生了语料中的所有的词；

上帝的这个坛子里面，骰子可以是无穷多个，有些类型的骰子数量多，有些类型的骰子少，所以从概率分布的角度看，坛子里面的骰子$\vec{p}$ 服从一个概率分布$p(\vec{p})$，这个分布称为参数$\vec{p}$ 的先验分布。

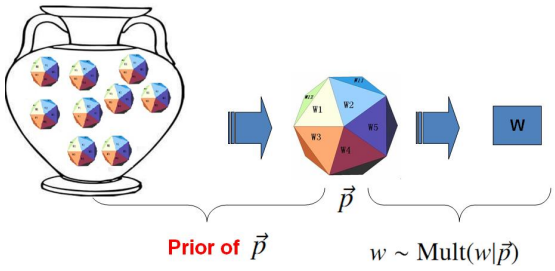


Figure 25: 贝叶斯观点下的Unigram Model

以上贝叶斯学派的游戏规则的假设之下，语料$\mathcal{W}$产生的概率如何计算呢？由于我们并不知道上帝到底用了哪个骰子$\vec{p}$,所以每个骰子都是可能被使用的，只是使用的概率由先验分布$p(\vec{p})$来决定。对每一个具体的骰子$\vec{p}$,由该骰子产生数据的概率是$p(\mathcal{W}|\vec{p})$, 所以最终数据产生的概率就是对每一个骰子$\vec{p}$上产生的数据概率进行积分累加求和

$$p(\mathcal{W}) = \int p(\mathcal{W}|\vec{p})p(\vec{p})d\vec{p}$$

在贝叶斯分析的框架下，此处先验分布$p(\vec{p})$ 就可以有很多种选择了，注意到

$$p(\vec{n}) = Mult(\vec{n}|\vec{p}, N)$$

实际上是在计算一个多项分布的概率，所以对先验分布的一个比较好的选择就是多项分布对应的共轭分布，即Dirichlet 分布

$$Dir(\vec{p}|\vec{\alpha}) = \frac{1}{\Delta(\vec{\alpha})}\prod_{k=1}^{V} p_k^{\alpha_k - 1} \quad \vec{\alpha} = (\alpha_1, \cdots, \alpha_V)$$

此处，$\Delta(\vec{\alpha})$ 就是归一化因子$Dir(\vec{\alpha})$，即

$$\Delta(\vec{\alpha}) = \int \prod_{k=1}^{V} p_k^{\alpha_k - 1} d\vec{p}.$$

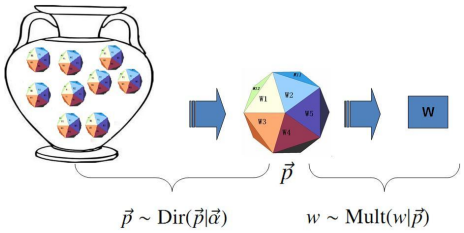


Figure 26: Dirichlet 先验下的Unigram Model

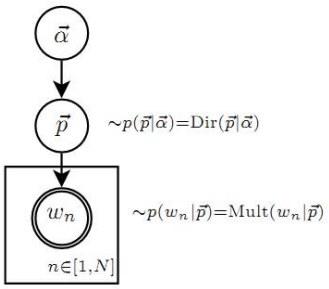


Figure 27: Unigram Model的概率图模型

回顾前一个小节介绍的Drichlet 分布的一些知识，其中很重要的一点就是

Dirichlet 先验+ 多项分布的数据$\longrightarrow$ 后验分布为Dirichlet 分布

$$Dir(\vec{p}|\vec{\alpha}) + MultCount(\vec{n}) = Dir(\vec{p}|\vec{\alpha}+\vec{n})$$

于是，在给定了参数$\vec{p}$的先验分布$Dir(\vec{p}|\vec{\alpha})$ 的时候，各个词出现频次的数据$\vec{n} \sim Mult(\vec{n}|\vec{p}, N)$ 为多项分布, 所以无需计算，我们就可以推出后验分布是

$$p(\vec{p}|\mathcal{W}, \vec{\alpha}) = Dir(\vec{p}|\vec{n}+\vec{\alpha}) = \frac{1}{\Delta(\vec{n}+\vec{\alpha})}\prod_{k=1}^{V} p_k^{n_k+\alpha_k-1} d\vec{p} \tag{23}$$

在贝叶斯的框架下，参数$\vec{p}$如何估计呢？由于我们已经有了参数的后验分布，所以合理的方式是使用后验分布的极大值点，或者是参数在后验分布下的平均值。在该文档中，我们取平均值作为参数的估计值。使用上个小节中(17)式的结论，由于$\vec{p}$ 的后验分布为$Dir(\vec{p}|\vec{n}+\vec{\alpha})$，于是

$$E(\vec{p}) = \Big(\frac{n_1+\alpha_1}{\sum_{i=1}^{V}(n_i+\alpha_i)}, \frac{n_2+\alpha_2}{\sum_{i=1}^{V}(n_i+\alpha_i)}, \cdots, \frac{n_V+\alpha_V}{\sum_{i=1}^{V}(n_i+\alpha_i)}\Big)$$

也就是说对每一个p_i, 我们用下式做参数估计

$$\hat{p_i} = \frac{n_i+\alpha_i}{\sum_{i=1}^{V}(n_i+\alpha_i)} \tag{24}$$

考虑到α_i 在Dirichlet 分布中的物理意义是事件的先验的伪计数，这个估计式子的含义是很直观的：每个参数的估计值是其对应事件的先验的伪计数和数据中的计数的和在整体计数中的比例。

进一步，我们可以计算出文本语料的产生概率为

$$\begin{aligned} p(\mathcal{W}|\vec{\alpha}) &= \int p(\mathcal{W}|\vec{p})p(\vec{p}|\vec{\alpha})d\vec{p} \\ &= \int \prod_{k=1}^{V} p_k^{n_k} Dir(\vec{p}|\vec{\alpha})d\vec{p} \\ &= \int \prod_{k=1}^{V} p_k^{n_k} \frac{1}{\Delta(\vec{\alpha})} \prod_{k=1}^{V} p_k^{\alpha_k - 1} d\vec{p} \\ &= \frac{1}{\Delta(\vec{\alpha})} \int \prod_{k=1}^{V} p_k^{n_k + \alpha_k - 1} d\vec{p} \\ &= \frac{\Delta(\vec{n} + \vec{\alpha})}{\Delta(\vec{\alpha})} \end{aligned} \tag{25}$$

0.5.2 Topic Model 和PLSA

以上Unigram Model 是一个很简单的模型，模型中的假设看起来过于简单，和人类写文章产生每一个词的过程差距比较大，有没有更好的模型呢？

我们可以看看日常生活中人是如何构思文章的。如果我们要写一篇文章，往往是先确定要写哪几个主题。譬如构思一篇自然语言处理相关的文章，可能40% 会谈论语言学、30% 谈论概率统计、20% 谈论计算机、还有10%谈论其它的主题：

- 说到语言学，我们容易想到的词包括：语法、句子、乔姆斯基、句法分析、主语...；
- 谈论概率统计，我们容易想到以下一些词: 概率、模型、均值、方差、证明、独立、马尔科夫链、...；
- 谈论计算机，我们容易想到的词是：内存、硬盘、编程、二进制、对象、算法、复杂度...；

我们之所以能马上想到这些词，是因为这些词在对应的主题下出现的概率很高。我们可以很自然的看到，一篇文章通常是由多个主题构成的、而每一个主题大概可以用与该主题相关的频率最高的一些词来描述。

以上这种直观的想法由Hoffmn 于1999 年给出的PLSA(Probabilistic Latent Semantic Analysis) 模型中首先进行了明确的数学化。Hoffman 认为一篇文

档(Document) 可以由多个主题(Topic) 混合而成，而每个Topic 都是词汇上的概率分布，文章中的每个词都是由一个固定的Topic 生成的。下图是英语中几个Topic 的例子。

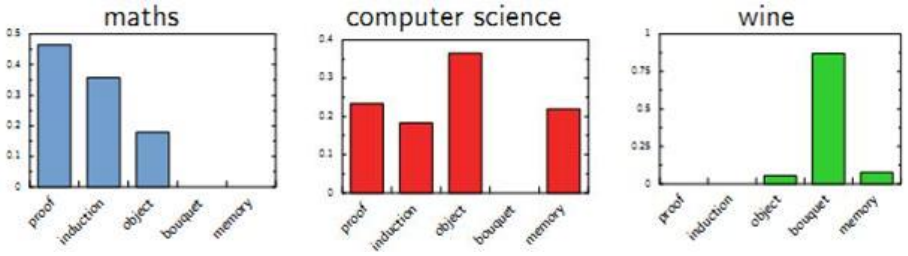


Figure 28: Topic 就是Vocab 上的概率分布

所有人类思考和写文章的行为都可以认为是上帝的行为，我们继续回到上帝的假设中，那么在PLSA 模型中，Hoffman 认为上帝是按照如下的游戏规则来生成文本的。

Game 11 PLSA Topic Model

1: 上帝有两种类型的骰子，一类是doc-topic 骰子,每个doc-topic 骰子有K 个面，每个面是一个topic 的编号；一类是topic-word 骰子，每个topic-word 骰子有V 个面，每个面对应一个词；

2: 上帝一共有K 个topic-word 骰子, 每个骰子有一个编号，编号从1 到K；

3: 生成每篇文档之前，上帝都先为这篇文章制造一个特定的doc-topic 骰子，然后重复如下过程生成文档中的词

- 投掷这个doc-topic 骰子,得到一个topic 编号z
- 选择K 个topic-word 骰子中编号为z的那个，投掷这个骰子，于是得到一个词

以上PLSA 模型的文档生成的过程可以图形化的表示为

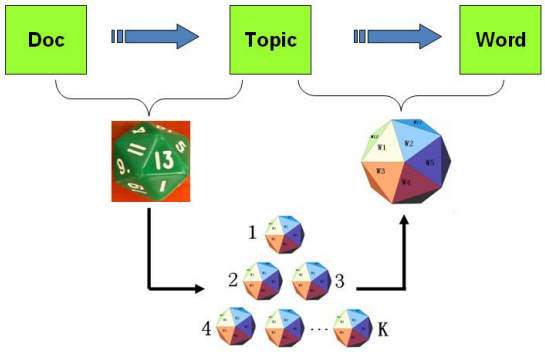


Figure 29: PLSA 模型的文档生成过程

我们可以发现在以上的游戏规则下，文档和文档之间是独立可交换的，同一个文档内的词也是独立可交换的，还是一个bag-of-words 模型。游戏中的K 个topic-word 骰子，我们可以记为$\vec{\varphi}_1, \cdots, \vec{\varphi}_K$, 对于包含$M$篇文档的语料$C = (d_1, d_2, \cdots, d_M)$ 中的每篇文档d_m，都会有一个特定的doc-topic骰子$\vec{\theta}_m$，所有对应的骰子记为$\vec{\theta}_1, \cdots, \vec{\theta}_M$。为了方便，我们假设每个词$w$ 都是一个编号，对应到topic-word 骰子的面。于是在PLSA 这个模型中，第m篇文档d_m 中的每个词的生成概率为

$$p(w|d_m) = \sum_{z=1}^{K} p(w|z)p(z|d_m) = \sum_{z=1}^{K} \varphi_{zw}\theta_{mz}$$

所以整篇文档的生成概率为

$$p(\vec{w}|d_m) = \prod_{i=1}^{n}\sum_{z=1}^{K} p(w_i|z)p(z|d_m) = \prod_{i=1}^{n}\sum_{z=1}^{K} \varphi_{zw_i}\theta_{dz}$$

由于文档之间相互独立，我们也容易写出整个语料的生成概率。求解PLSA 这个Topic Model 的过程汇总，模型参数并容易求解，可以使用著名的EM 算法进行求得局部最优解，由于该模型的求解并不是本文的介绍要点，有兴趣的同学参考Hoffman 的原始论文，此处略去不讲。

0.6 LDA 文本建模

0.6.1 游戏规则

对于上述的PLSA 模型，贝叶斯学派显然是有意见的，doc-topic 骰子$\vec{\theta}_m$和topic-word 骰子$\vec{\varphi}_k$都是模型中的参数，参数都是随机变量，怎么能没有先验

分布呢？于是，类似于对Unigram Model 的贝叶斯改造，我们也可以如下在两个骰子参数前加上先验分布从而把PLSA 对应的游戏过程改造为一个贝叶斯的游戏过程。由于$\vec{\varphi}_k$和$\vec{\theta}_m$都对应到多项分布，所以先验分布的一个好的选择就是Drichlet 分布，于是我们就得到了LDA(Latent Dirichlet Allocation)模型。

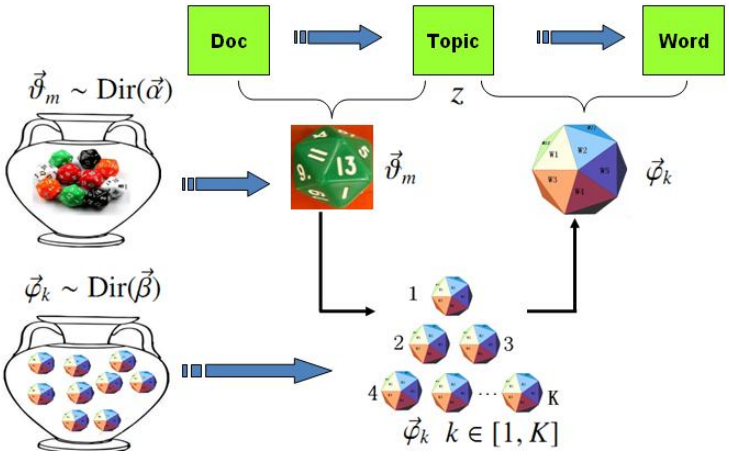
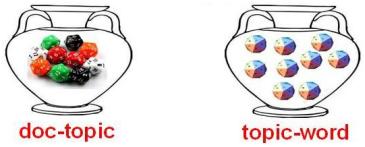


Figure 30: LDA模型

在LDA 模型中, 上帝是按照如下的规则玩文档生成的游戏的

Game 12 LDA Topic Model

1: 上帝有两大坛子的骰子，第一个坛子装的是doc-topic 骰子,第二个坛子装的是topic-word 骰子；



2: 上帝随机的从第二个坛子中独立的抽取了K 个topic-word 骰子，编号为1到K；

3: 每次生成一篇新的文档前，上帝先从第一个坛子中随机抽取一个doc-topic 骰子，然后重复如下过程生成文档中的词

- 投掷这个doc-topic 骰子,得到一个topic 编号z
- 选择K 个topic-word 骰子中编号为z的那个，投掷这个骰子，于是得到一个词

假设语料库中有M 篇文档，所有的的word和对应的topic 如下表示

$$\vec{\mathbf{w}} = (\vec{w}_1, \cdots, \vec{w}_M)$$
$$\vec{\mathbf{z}} = (\vec{z}_1, \cdots, \vec{z}_M)$$

其中，$\vec{w}_m$ 表示第m 篇文档中的词，$\vec{z}_m$ 表示这些词对应的topic 编号。

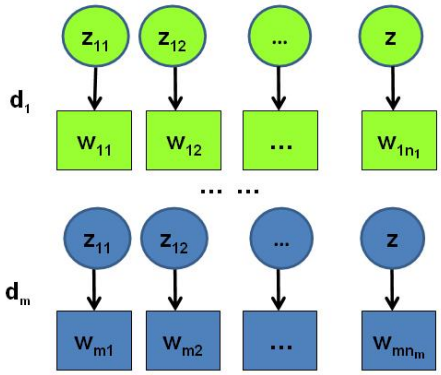


Figure 31: 语料生成过程中的word 和topic

0.6.2 物理过程分解

使用概率图模型表示，LDA 模型的游戏过程如图所示。

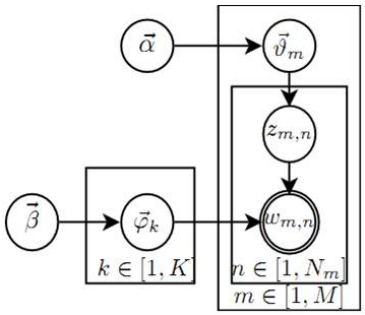


Figure 32: LDA图模型表示

这个概率图可以分解为两个主要的物理过程：

1. $\vec{\alpha} \to \vec{\theta}_m \to z_{m,n}$, 这个过程表示在生成第$m$ 篇文档的时候，先从第一个坛子中抽了一个doc-topic 骰子$\vec{\theta}_m$, 然后投掷这个骰子生成了文档中第n 个词的topic编号$z_{m,n}$；

2. $\vec{\beta} \rightarrow \vec{\varphi}_k \rightarrow w_{m,n}|k = z_{m,n}$, 这个过程表示用如下动作生成语料中第$m$篇文档的第$n$个词：在上帝手头的$K$ 个topic-word 骰子$\vec{\varphi}_k$ 中，挑选编号为$k = z_{m,n}$的那个骰子进行投掷，然后生成word $w_{m,n}$；

理解LDA最重要的就是理解这两个物理过程。LDA 模型在基于K 个topic生成语料中的M 篇文档的过程中，由于是bag-of-words 模型，有一些物理过程是相互独立可交换的。由此，LDA 生成模型中，M 篇文档会对应于M 个独立的Dirichlet-Multinomial 共轭结构；K 个topic 会对应于K 个独立的Dirichlet-Multinomial 共轭结构。所以理解LDA 所需要的所有数学就是理解Dirichlet-Multiomail 共轭，其它都就是理解物理过程。现在我们进入细节，来看看LDA模型是如何被分解为$M + K$ 个Dirichlet-Multinomial 共轭结构的。

由第一个物理过程，我们知道$\vec{\alpha} \rightarrow \vec{\theta}_m \rightarrow \vec{z}_m$ 表示生成第m 篇文档中的所有词对应的topics，显然$\vec{\alpha} \rightarrow \vec{\theta}_m$ 对应于Dirichlet 分布，$\vec{\theta}_m \rightarrow \vec{z}_m$ 对应于Multinomial 分布，所以整体是一个Dirichlet-Multinomial 共轭结构；

$$\vec{\alpha} \underbrace{\longrightarrow}_{Dirichlet} \vec{\theta}_m \underbrace{\longrightarrow}_{Multinomial} \vec{z}_m$$

前文介绍Bayesian Unigram Model 的小节中我们对Dirichlet-Multinomial 共轭结构做了一些计算。借助于该小节中的(25)式，我们可以得到

$$p(\vec{z}_m|\vec{\alpha}) = \frac{\Delta(\vec{n}_m + \vec{\alpha})}{\Delta(\vec{\alpha})}$$

其中$\vec{n}_m = (n_m^{(1)}, \cdots, n_m^{(K)})$，$n_m^{(k)}$ 表示第m篇文档中第k 个topic 产生的词的个数。进一步，利用Dirichlet-Multiomial 共轭结构，我们得到参数$\vec{\theta}_m$ 的后验分布恰好是

$$Dir(\vec{\theta}_m|\vec{n}_m + \vec{\alpha}).$$

由于语料中M篇文档的topics 生成过程相互独立，所以我们得到M 个相互独立的Dirichlet-Multinomial 共轭结构，从而我们可以得到整个语料中topics生成概率

$$\begin{aligned} p(\vec{\mathbf{z}}|\vec{\alpha}) &= \prod_{m=1}^{M} p(\vec{z}_m|\vec{\alpha}) \\ &= \prod_{m=1}^{M} \frac{\Delta(\vec{n}_m + \vec{\alpha})}{\Delta(\vec{\alpha})} \end{aligned} \tag{26}$$

目前为止，我们由M篇文档得到了M 个Dirichlet-Multinomial 共轭结构，还有额外K 个Dirichlet-Multinomial 共轭结构在哪儿呢？在上帝按照之前的规

则玩LDA 游戏的时候，上帝是先完全处理完成一篇文档，再处理下一篇文档。文档中每个词的生成都要抛两次骰子，第一次抛一个doc-topic骰子得到topic, 第二次抛一个topic-word骰子得到word，每次生成每篇文档中的一个词的时候这两次抛骰子的动作是紧邻轮换进行的。如果语料中一共有N 个词，则上帝一共要抛$2N$次骰子，轮换的抛doc-topic骰子和topic-word骰子。但实际上有一些抛骰子的顺序是可以交换的，我们可以等价的调整$2N$次抛骰子的次序：前N次只抛doc-topic骰子得到语料中所有词的topics,然后基于得到的每个词的topic 编号，后N次只抛topic-word骰子生成N 个word。于是上帝在玩LDA 游戏的时候，可以等价的按照如下过程进行：

Game 13 LDA Topic Model 2

1: 上帝有两大坛子的骰子，第一个坛子装的是doc-topic 骰子,第二个坛子装的是topic-word 骰子；
2: 上帝随机的从第二个坛子中独立的抽取了K 个topic-word 骰子，编号从1到K；
3: 每次生成一篇新的文档前，上帝先从第一个坛子中随机抽取一个doc-topic骰子，然后重复投掷这个doc-topic 骰子,为每个词生成一个topic 编号z；重复如上过程处理每篇文档，生成语料中每个词的topic 编号，但是词尚未生成；
4: 从头到尾，对语料中的每篇文档中的每个topic 编号z, 选择K 个topic-word骰子中编号为z的那个，投掷这个骰子，于是生成对应的word；

以上游戏是先生成了语料中所有词的topic, 然后对每个词在给定topic 的条件下生成word。在语料中所有词的topic 已经生成的条件下，任何两个word 的生成动作都是可交换的。于是我们把语料中的词进行交换，把具有相同topic 的词放在一起

$$\vec{\mathbf{w}}' = (\vec{w}_{(1)}, \cdots, \vec{w}_{(K)})$$
$$\vec{\mathbf{z}}' = (\vec{z}_{(1)}, \cdots, \vec{z}_{(K)})$$

其中，$\vec{w}_{(k)}$ 表示这些词都是由第k 个topic 生成的，$\vec{z}_{(k)}$ 对应于这些词的topic编号，所以$\vec{z}_{(k)}$中的分量都是k。

对应于概率图中的第二个物理过程$\vec{\beta} \to \vec{\varphi}_k \to w_{m,n}|k = z_{m,n}$，在$k = z_{m,n}$的限制下，语料中任何两个由topic k 生成的词都是可交换的，即便他们不再同一个文档中，所以我们此处不再考虑文档的概念，转而考虑由同一个topic生成的词。考虑如下过程$\vec{\beta} \to \vec{\varphi}_k \to \vec{w}_{(k)}$ ，容易看出，此时$\vec{\beta} \to \vec{\varphi}_k$ 对应

于Dirichlet 分布，$\vec{\varphi}_k \rightarrow \vec{w}_{(k)}$ 对应于Multinomial 分布，所以整体也还是一个Dirichlet-Multinomial 共轭结构；

$$\vec{\beta} \underbrace{\longrightarrow}_{Dirichlet} \vec{\varphi}_k \underbrace{\longrightarrow}_{Multinomial} \vec{w}_{(k)}$$

同样的借助于(25)式，我们可以得到

$$p(\vec{w}_{(k)}|\vec{\beta}) = \frac{\Delta(\vec{n}_k + \vec{\beta})}{\Delta(\vec{\beta})}$$

其中$\vec{n}_k = (n_k^{(1)}, \cdots, n_k^{(V)})$，$n_k^{(t)}$ 表示第k 个topic 产生的词中word t的个数。进一步，利用Dirichlet-Multiomial 共轭结构，我们得到参数$\vec{\varphi}_k$ 的后验分布恰好是

$$Dir(\vec{\varphi}_k|\vec{n}_k + \vec{\beta}).$$

而语料中K个topics 生成words 的过程相互独立，所以我们得到K 个相互独立的Dirichlet-Multinomial 共轭结构，从而我们可以得到整个语料中词生成概率

$$\begin{aligned} p(\vec{\mathbf{w}}|\vec{\mathbf{z}}, \vec{\beta}) &= p(\vec{\mathbf{w}}'|\vec{\mathbf{z}}', \vec{\beta}) \\ &= \prod_{k=1}^{K} p(\vec{w}_{(k)}|\vec{z}_{(k)}, \vec{\beta}) \\ &= \prod_{k=1}^{K} \frac{\Delta(\vec{n}_k + \vec{\beta})}{\Delta(\vec{\beta})} \end{aligned} \tag{27}$$

结合(26) 和(27) 于是我们得到

$$\begin{aligned} p(\vec{\mathbf{w}}, \vec{\mathbf{z}}|\vec{\alpha}, \vec{\beta}) &= p(\vec{\mathbf{w}}|\vec{\mathbf{z}}, \vec{\beta}) p(\vec{\mathbf{z}}|\vec{\alpha}) \\ &= \prod_{k=1}^{K} \frac{\Delta(\vec{n}_k + \vec{\beta})}{\Delta(\vec{\beta})} \prod_{m=1}^{M} \frac{\Delta(\vec{n}_m + \vec{\alpha})}{\Delta(\vec{\alpha})} \end{aligned} \tag{28}$$

此处的符号表示稍微不够严谨, 向量$\vec{n}_k$, $\vec{n}_m$ 都用n 表示，主要通过下标进行区分，k 下标为topic 编号, m 下标为文档编号。

0.6.3 Gibbs Sampling

有了联合分布$p(\vec{\mathbf{w}}, \vec{\mathbf{z}})$, 万能的MCMC 算法就可以发挥作用了！于是我们可以考虑使用Gibbs Sampling 算法对这个分布进行采样。当然由于$\vec{\mathbf{w}}$ 是观测到的已知数据，只有$\vec{\mathbf{z}}$是隐含的变量，所以我们真正需要采样的是分布$p(\vec{\mathbf{z}}|\vec{\mathbf{w}})$。

在Gregor Heinrich 那篇很有名的LDA 模型科普文章*Parameter estimation for text analysis* 中，是基于(28) 式推导Gibbs Sampling 公式的。此小节中我们使用不同的方式，主要是基于Dirichlet-Multinomial 共轭来推导Gibbs Sampling 公式，这样对于理解采样中的概率物理过程有帮助。

语料库$\vec{\mathbf{z}}$ 中的第i个词对应的topic 我们记为z_i, 其中$i = (m, n)$是一个二维下标，对应于第m篇文档的第n个词，我们用$\neg i$ 表示去除下标为i的词。那么按照Gibbs Sampling 算法的要求，我们要求得任一个坐标轴i 对应的条件分布$p(z_i = k|\vec{\mathbf{z}}_{\neg i}, \vec{\mathbf{w}})$ 。假设已经观测到的词$w_i = t$, 则由贝叶斯法则，我们容易得到

$$p(z_i = k|\vec{\mathbf{z}}_{\neg i}, \vec{\mathbf{w}}) \propto p(z_i = k, w_i = t|\vec{\mathbf{z}}_{\neg i}, \vec{\mathbf{w}}_{\neg i})$$

由于$z_i = k, w_i = t$ 只涉及到第m 篇文档和第k个topic，所以上式的条件概率计算中, 实际上也只会涉及到如下两个Dirichlet-Multinomial 共轭结构

1. $\vec{\alpha} \to \vec{\theta}_m \to \vec{z}_m$
2. $\vec{\beta} \to \vec{\varphi}_k \to \vec{w}_{(k)}$

其它的$M + K - 2$ 个Dirichlet-Multinomial 共轭结构和$z_i = k, w_i = t$是独立的。

由于在语料去掉第i 个词对应的(z_i, w_i)，并不改变我们之前讨论的$M + K$个Dirichlet-Multinomial 共轭结构，只是某些地方的计数会减少。所以$\vec{\theta}_m, \vec{\varphi}_k$的后验分布都是Dirichlet:

$$p(\vec{\theta}_m|\vec{\mathbf{z}}_{\neg i}, \vec{\mathbf{w}}_{\neg i}) = Dir(\vec{\theta}_m|\vec{n}_{m,\neg i} + \vec{\alpha})$$
$$p(\vec{\varphi}_k|\vec{\mathbf{z}}_{\neg i}, \vec{\mathbf{w}}_{\neg i}) = Dir(\vec{\varphi}_k|\vec{n}_{k\neg i} + \vec{\beta})$$

使用上面两个式子，把以上想法综合一下，我们就得到了如下的Gibbs Sam-

pling 公式的推导

$$
\begin{aligned}
p(z_i = k|\vec{\mathbf{z}}_{\neg i}, \vec{\mathbf{w}}) &\propto p(z_i = k, w_i = t|\vec{\mathbf{z}}_{\neg i}, \vec{\mathbf{w}}_{\neg i}) \\
&= \int p(z_i = k, w_i = t, \vec{\theta}_m, \vec{\varphi}_k|\vec{\mathbf{z}}_{\neg i}, \vec{\mathbf{w}}_{\neg i}) d\vec{\theta}_m d\vec{\varphi}_k \\
&= \int p(z_i = k, \vec{\theta}_m|\vec{\mathbf{z}}_{\neg i}, \vec{\mathbf{w}}_{\neg i}) \cdot p(w_i = t, \vec{\varphi}_k|\vec{\mathbf{z}}_{\neg i}, \vec{\mathbf{w}}_{\neg i}) d\vec{\theta}_m d\vec{\varphi}_k \\
&= \int p(z_i = k|\vec{\theta}_m) p(\vec{\theta}_m|\vec{\mathbf{z}}_{\neg i}, \vec{\mathbf{w}}_{\neg i}) \cdot p(w_i = t|\vec{\varphi}_k) p(\vec{\varphi}_k|\vec{\mathbf{z}}_{\neg i}, \vec{\mathbf{w}}_{\neg i}) d\vec{\theta}_m d\vec{\varphi}_k \\
&= \int p(z_i = k|\vec{\theta}_m) Dir(\vec{\theta}_m|\vec{n}_{m,\neg i} + \vec{\alpha}) d\vec{\theta}_m \\
&\quad \cdot \int p(w_i = t|\vec{\varphi}_k) Dir(\vec{\varphi}_k|\vec{n}_{k,\neg i} + \vec{\beta}) d\vec{\varphi}_k \\
&= \int \theta_{mk} Dir(\vec{\theta}_m|\vec{n}_{m,\neg i} + \vec{\alpha}) d\vec{\theta}_m \cdot \int \varphi_{kt} Dir(\vec{\varphi}_k|\vec{n}_{k,\neg i} + \vec{\beta}) d\vec{\varphi}_k \\
&= E(\theta_{mk}) \cdot E(\varphi_{kt}) \\
&= \hat{\theta}_{mk} \cdot \hat{\varphi}_{kt}
\end{aligned}
$$

以上推导估计是整篇文章中最复杂的数学了，表面上看上去复杂，但是推导过程中的概率物理意义是简单明了的：$z_i = k, w_i = t$的概率只和两个Dirichlet-Multinomail 共轭结构关联。而最终得到的$\hat{\theta}_{mk}, \hat{\varphi}_{kt}$ 就是对应的两个Dirichlet 后验分布在贝叶斯框架下的参数估计。借助于前面介绍的Dirichlet参数估计的公式(24)，我们有

$$
\begin{aligned}
\hat{\theta}_{mk} &= \frac{n_{m,\neg i}^{(k)} + \alpha_k}{\sum_{k=1}^{K}(n_{m,\neg i}^{(t)} + \alpha_k)} \\
\hat{\varphi}_{kt} &= \frac{n_{k,\neg i}^{(t)} + \beta_t}{\sum_{t=1}^{V}(n_{k,\neg i}^{(t)} + \beta_t)}
\end{aligned}
$$

于是，我们最终得到了LDA 模型的Gibbs Sampling 公式

$$
p(z_i = k|\vec{\mathbf{z}}_{\neg i}, \vec{\mathbf{w}}) \propto \frac{n_{m,\neg i}^{(k)} + \alpha_k}{\sum_{k=1}^{K}(n_{m,\neg i}^{(t)} + \alpha_k)} \cdot \frac{n_{k,\neg i}^{(t)} + \beta_t}{\sum_{t=1}^{V}(n_{k,\neg i}^{(t)} + \beta_t)} \tag{29}
$$

这个公式是很漂亮的，右边其实就是$p(topic|doc) \cdot p(word|topic)$，这个概率其实是$doc \to topic \to word$ 的路径概率，由于topic 有K 个，所以Gibbs Sampling 公式的物理意义其实就是在这K 条路径中进行采样。

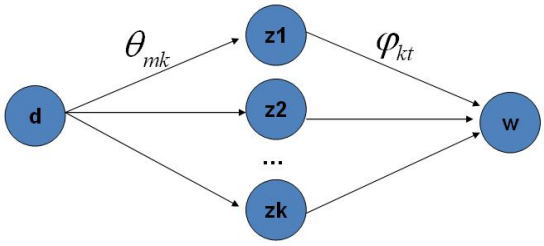


Figure 33: doc-topic-word 路径概率

0.6.4 Training and Inference

有了LDA 模型，当然我们的目标有两个

- 估计模型中的参数$\vec{\varphi}_1, \cdots, \vec{\varphi}_K$ 和$\vec{\theta}_1, \cdots, \vec{\theta}_M$;
- 对于新来的一篇文档doc_{new}，我们能够计算这篇文档的topic 分布$\vec{\theta}_{new}$。

有了Gibbs Sampling 公式，我们就可以基于语料训练LDA 模型，并应用训练得到的模型对新的文档进行topic 语义分析。训练的过程就是通过Gibbs Sampling 获取语料中的(z, w) 的样本，而模型中的所有的参数都可以基于最终采样得到的样本进行估计。训练的流程很简单:

Algorithm 14 LDA Training

1: 随机初始化：对语料中每篇文档中的每个词w，随机的赋一个topic 编号z;
2: 重新扫描语料库，对每个词w, 按照Gibbs Sampling 公式重新采样它的topic，在语料中进行更新;
3: 重复以上语料库的重新采样过程直到Gibbs Sampling 收敛;
4: 统计语料库的topic-word 共现频率矩阵，该矩阵就是LDA的模型;

对于Gibbs Sampling 算法实现的细节，请参考Gregor Heinrich 的*Parameter estimation for text analysis* 中对算法的描述，以及PLDA 的代码实现，此处不再赘述。

由这个topic-word 频率矩阵我们可以计算每一个$p(word|topic)$概率，从而算出模型参数$\vec{\varphi}_1, \cdots, \vec{\varphi}_K$, 这就是上帝用的$K$ 个topic-word 骰子。当然，语料中的文档对应的骰子参数$\vec{\theta}_1, \cdots, \vec{\theta}_M$ 在以上训练过程中也是可以计算出来的，只要在Gibbs Sampling 收敛之后，统计每篇文档中的topic 的频率分布，我们就可以计算每一个$p(topic|doc)$ 概率，于是就可以计算出每一个$\vec{\theta}_m$。由于参数$\vec{\theta}_m$是和训练语料中的每篇文档相关的，对于我们理解新的文档并无用处，所以工

程上最终存储LDA 模型时候一般没有必要保留。通常，在LDA 模型训练的过程中，我们是取Gibbs Sampling 收敛之后的n 个迭代的结果进行平均来做参数估计，这样模型质量更高。

有了LDA 的模型，对于新来的文档doc_{new}, 我们如何做该文档的topic 语义分布的计算呢？基本上inference 的过程和training 的过程完全类似。对于新的文档，我们只要认为Gibbs Sampling 公式中的$\hat{\varphi}_{kt}$ 部分是稳定不变的，是由训练语料得到的模型提供的，所以采样过程中我们只要估计该文档的topic 分布$\vec{\theta}_{new}$就好了。

Algorithm 15 LDA Inference

1: 随机初始化：对当前文档中的每个词w，随机的赋一个topic 编号z；
2: 重新扫描当前文档，按照Gibbs Sampling 公式，对每个词w, 重新采样它的topic;
3: 重复以上过程直到Gibbs Sampling 收敛；
4: 统计文档中的topic分布，该分布就是$\vec{\theta}_{new}$

0.7 后记

LDA 对于专业做机器学习的兄弟而言，只能算是一个简单的Topic Model。但是对于互联网中做数据挖掘、语义分析的工程师，LDA 的门槛并不低。LDA 典型的属于这样一种机器学习模型：要想理解它，需要比较多的数学背景，要在工程上进行实现，却相对简单。Gregor Heinrich 的LDA 模型科普文章*Parameter estimation for text analysis* 写得非常的出色，这是学习LDA 的必看文章。不过即便是这篇文章，对于工程师也是有门槛的。我写的这个科普最好对照Gregor Heinrich 的这篇文章来看，我用的数学符号也是尽可能和这篇文章保持一致。

这份LDA 科普是基于给组内兄弟做报告的ppt 整理而成的，说是科普其实也不简单，涉及到的数学还是太多。在工业界也混了几年，经常感觉到工程师对于学术界的玩的模型有很强的学习和尝试的欲望，只是学习成本往往太高。所以我写LDA 的初衷就是写给工业界的工程师们看的，希望把学术界玩的一些模型用相对通俗的方式介绍给工程师；如果这个科普对于读研究生的一些兄弟姐妹也有所启发，只能说那是一个side effect :-)。

我个人很喜欢LDA ，它是在文本建模中一个非常优雅的模型，相比于很多其它的贝叶斯模型，LDA 在数学推导上简洁优美。学术界自2003 年以来也输出了很多基于LDA 的Topic Model 的变体，要想理解这些更加高

级的Topic Model, 首先需要很好的理解标准的LDA 模型。在工业界，Topic Model 在Google、Baidu 等大公司的产品的语义分析中都有着重要的应用；所以Topic Model 对于工程师而言，这是一个很有应用价值、值得学习的模型。我接触Topic Model 的时间不长，主要是由于2年前和PLDA 的作者Wangyi 一起合作的过程中，从他身上学到了很多Topic Model 方面的知识。关于LDA 的相关知识，其实可以写的还有很多：如何提高LDA Gibbs Sampling 的速度、如何优化超参数、如何做大规模并行化、LDA 的应用、LDA 的各种变体..... 不过我的主要目标还是科普如何理解标准的LDA 模型。

学习一个模型的时候我喜欢追根溯源，常常希望把模型中的每一个数学推导的细节搞明白，把公式的物理意义想清楚，不过数学推导本身并不是我想要的，把数学推导还原为物理过程才是我乐意做的事。最后引用一下物理学家费曼的名言结束LDA 的数学科普：

What I cannot create, I do not understand.

— Richard Feynman