低动能机器人技术

走路、跑步、游泳、 飞行和操纵的算法

吕斯-特德雷克

最后修改 2021-6-13.
如何引用这些笔记，使用注释，并给予反馈。

注意：这些是用于在麻省理工学院教授的课程的工作笔记。 它们将在2021年春季学期中被更新。 讲座视频可在YouTube上找到。

章节 13

反馈运动规划

低动能机器人技术

走路、跑步、游泳、飞行和操纵的算法

吕斯-特德雷克

最后修改 2021-6-13.

注意：这些是用于在麻省理工学院教授的课程的工作笔记。它们将在2021年春季学期中被更新。讲座视频可在YouTube上找到。

上一章　　目录　　下一章

章节 14

稳健和随机控制

到目前为止，我们在笔记中只关注已知的、确定性的系统。在本章中，我们将开始考虑不确定性。这种不确定性有多种形式.....我们可能不知道管理方程（例如关节中的摩擦系数），我们的机器人可能在未知的地形上行走，受到未知的干扰，甚至是在捡拾未知的物体。有许多数学框架可以考虑这种不确定性；为了我们的目的，本章将把我们的想法概括为这样的方程。

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{w}, t) \quad \text{或} \quad \mathbf{x}[n+1] = \mathbf{f}(\mathbf{x}[n], \mathbf{u}[n], \mathbf{w}[n], n)。$$

其中$\mathbf{w}$是一个新的*随机*输入信号，用于捕捉所有这些潜在的可变性。尽管在连续时间内工作，并将$\mathbf{w}(t)$作为一个连续时间的随机信号（比方说维纳过程）是肯定可行的，但将$\mathbf{w}[n]$作为一个离散时间的随机信号来工作，在符号学上更为简单。出于这个原因，我们在这一章将把注意力放在离散时间系统上。

为了模拟这种形式的方程，或者针对它们设计控制器，我们需要确定产生$\mathbf{w}[n]$的随机过程。典型的做法是假设$\mathbf{w}[n]$的值是独立和同分布的（i.i.d.），也就是说，当$i \neq j$时，$\mathbf{w}[i]$和$\mathbf{w}[j]$是不相关的。因此，我们通常通过概率密度$p_{\mathbf{w}}(\mathbf{w}[n])$来定义我们的分布。这并不像它听起来那样具有局限性.....如果我们希望
对待时间相关的噪声（例如 "彩色噪声"），我们方程的格式足够丰富，可以通过添加额外的状态变量来支持；我们将在下面给出一个用于模拟阵风的 "增白Ølter "的例子。我们现在要考虑方程中的另一个随机性来源是初始条件中的随机性；我们将类似地定义一个概率密度$p_{\mathbf{x}}(\mathbf{x}[0])$。

这个建模框架足够丰富，可以让我们传达关键的想法；但对于我感兴趣的所有系统来说，它还不够完善。在DRAKE中，我们花了更多的时间来支持随机系统的一般情况。这包括对系统参数进行建模，这些参数在模型初始化时是随机抽取的，但在模拟过程中是固定的；将这些参数作为没有动态的额外状态变量进行建模是可能的，但不足够。在其他问题中，甚至*状态向量*的*维度也*可能在不同的现实中发生变化。

问题!例如，考虑一下机器人操纵水槽中随机数量的餐具的情况。我不知道有多少控制公式能很好地处理这种类型的随机性，我认为这是一个最优先考虑的问题！（我们将在输出反馈一章中开始解决这个问题）。(我们将在输出反馈一章中开始解决这个问题）。

粗略地说，我将把 "随机控制 "称为合成控制器的学科，这些控制器控制着方程的概率演变。"随机最优控制 "定义了一个成本函数（现在是一个随机变量），并试图找到优化某些指标的控制器，如预期成本。 当我们使用 "鲁棒控制 "这个术语时，我们通常指的是一类技术，它试图保证最坏情况下的性能，或对输入的随机性对输出的随机性的影响的最坏情况下的约束。有趣的是，对于许多鲁棒控制公式，我们并不试图知道**x**[0]和**w**[n]的精确概率分布，而只是确定可能的值的*集合*，即
他们可以采取。这种建模很强大，但可能导致保守的控制器和对性能的悲观估计。

我的目标是对一些主要观点进行相对可消耗的调查，这一章也许比其他任何一章都更重要。有人说，"稳健控制是加密的"（因为你需要知道秘密代码才能进入）。稳健控制界的文化是利用高能的数学，有时以提供更简单的解释为代价。我认为这是不幸的，因为机器人学和机器学习将从与这些工具更丰富的联系中受益，而且也许注定要重新创造许多工具。

稳健控制的经典参考文献是[1].[2]有一个处理方法，它在时域和通过Riccati方程做了更多的发展。

14.1 有限马尔可夫决策过程

我们已经快速预览了随机最优控制中的一种情况，它特别容易。Ønite Markov Decision Processes（MDPs）。

14.2 线性最优控制

14.2.1 分析报告

常见的Lyapunov函数

我们已经看到了线性系统鲁棒性分析的一个很好的例子，当时我们写了一个小的优化，为不确定的线性系统找到一个共同的李亚普诺夫函数。这个例子研究了动力学**x**[n+1]=**Ax**[n]，其中**A**的系数是未知的，但却是有界的。

我们还看到，基本上同样的技术可以用来证明对干扰的稳定性，例如。

$$\mathbf{x}[n+1] = \mathbf{A}\mathbf{x}[n] + \mathbf{w}[n], \qquad \mathbf{w}[n] \in \mathbb{W}。$$

其中，W描述了一些有界的可能的不确定性集合。为了与凸优化兼容，我们通常选择将W描述为一个椭圆体或一个凸多角形[3].但我们要记住一个非常重要的问题：在这种有加性干扰的情况下，我们不能再期望系统稳定地收敛到原点。我们之前的例子只是用普通的李亚普诺夫函数来证明一个区域的*不变性*（如果我在某个区域内开始，那么我就不会离开）。

L_2增益

在某种意义上，上面的共-利亚普诺夫分析可能是错误的分析

对于线性系统（也许还有其他系统）。假设干扰是有界限的，这可能是不合理的。此外，我们知道对输入（包括干扰输入）的响应是线性的，所以我们期望与未受干扰的情况相比，**x**的偏差的大小与干扰的大小**w**成正比。

通常情况下，这是用标量 "L_2增益"，γ来完成的，定义为：。

$$\text{争论的焦点}_{\gamma} \quad \text{课题组} \quad \frac{\int_0^T |\mathbf{x}(t)|^2 dt}{\mathbf{w}(\cdot)\in \int_0^T |\mathbf{w}(t)|^2 dt \le \infty} \le \gamma^2 \quad \text{，或}$$

$$\text{争论的焦点}_{\gamma} \quad \text{课题组} \quad \frac{\sum_0^N |\mathbf{x}[n]|^2}{\mathbf{w}[\cdot]\in \sum_n |\mathbf{w}[n]|^2 \le \infty \; \sum_0^N |\mathbf{w}[n]|^2} \le \gamma^2 .$$

L_2增益 "的名称来自于对信号$\mathbf{w}(t)$和$\mathbf{x}(t)$使用ℓ_2准则，它只被假定为Ønite。

更多时候，这些收益不是直接用$\mathbf{x}[n]$来写，而是用一些 "性能输出 "$\mathbf{z}[n]$来写。例如，如果想把二次调节器目标的成本约束为干扰大小的函数，我们可以最小化

$$\text{闽}_{\gamma} \quad \text{受制于} \quad \text{以下情况} \frac{\sum^N |\mathbf{z}[n]|^2}{{}^2\mathbf{z}[n]= \; \mathbf{w}[n] \sum^N_0 |\mathbf{w}[n]|^2} \le \gamma, \quad \left[\begin{array}{c} 1\sqrt{} \\ \mathbf{Qx}[n] \\ \sqrt{\mathbf{Ru}[n]} \end{array} \right].$$

这是一个简单但重要的想法，理解它是理解围绕稳健控制的语言的关键。特别是系统的H_2规范（从输入**w**到输出**z**）是脉冲响应的能量；当**z**被选为代表上述的二次调节器成本时，它对应于预期的LQR成本。系统的H_∞规范（从**w**到**z**）是脉冲响应的最大奇异值。
传递函数；它对应于L_2增益。

小增益定理

即将推出...

耗散不等式

即将推出...见，例如，[4] 或 [5，第2章]。

14.2.2 H_2设计

14.2.3 H_∞设计

14.2.4 线性指数-二次方高斯（LEQG）。

[6]观察到，使目标最小化也是直截了当的。

$$J = E\left[\prod_{n=0}^{\infty} {}^{exT}[n]\mathbf{Qx}[n]^{euT}[n]\mathbf{Ru}[n]\right] = E\left\{e^{\sum_{n=0}^{\infty} {}^{xT}[n]\mathbf{Qx}[n]+{}^{uT}[n]\mathbf{Ru}[n]}\right\}。$$

通过观察成本与对数J呈单调关系，因此具有相同的最小值，可以得出$\mathbf{Q}={}^{QT}\ge\mathbf{0}$，$\mathbf{R}={}^{RT}>0$的结论（这一招也构成了 "对数 "的基础）。
"几何编程"[7]).这被称为 "线性指数-二次高斯"（LEG或LEQG），对于问题的确定性版本（没有过程或测量噪声），解决方案与LQR问题相同；它没有增加新的建模能力。但在有噪声的情况下，LEQG的最优控制器是不同的

来自LQG控制器；它们明确地取决于噪声的协方差。

[8]对这个框架进行了广泛的处理；几乎所有来自LQR/LQG的分析（包括Riccati方程、Hamilton公式等）都有类似的指数成本版本，他认为LQG和H-inØnity控制可以（应该）被理解为这种方法的特例。

14.2.5 自适应控制

对H_2型最优控制的标准批评是，最小化期望值并不允许对性能有任何保证。对H_∞型最优控制的标准批评是，它关注的是最坏的情况，因此可能是
保守的，特别是因为先验选择的可能干扰的分布可能是不必要的保守的。人们可能希望，如果我们能够在线更新我们的不确定性模型，适应我们实际收到的扰动的统计数据，我们可以重新获得一些这样的性能。这就是自适应控制的目标之一。

在线自适应控制的一个基本问题是探索和利用之间的权衡。一些输入可能会促使系统迅速建立更准确的动态/不确定性模型，这可能会导致更好的性能。但是，我们怎样才能将这种权衡正式化呢？

在机器学习中，在（上下文）多臂匪徒问题的设置上，在这个挑战上已经有了一些不错的进展。为了我们的目的，你可以把土匪问题看作是最优控制问题的一个限制性案例，在这个问题上不存在动力（一个控制行动的效果不会影响到下一个控制行动的结果）。在这个更简单的环境中，在线优化社区已经开发了基于最小化*遗憾概念的*探索--通常是我的在线算法所取得的性能与如果我在开始之前就知道我的实验数据所能取得的性能之间的累积差异。这就导致了一些方法的出现，这些方法利用了诸如上界（UCB）等概念，最近还使用了最小二乘法的上界[9] 来提供遗憾的界限。

在过去的几年里，我们看到这些结果被转化为线性最优控制的设定。

14.2.6 结构性的不确定性

14.2.7 线性参数变化（LPV）控制

14.3 轨迹优化

14.3.1 蒙特卡洛轨迹优化

14.3.2 迭代式H_2

14.3.3 有限时间（可达性）分析

即将推出...

14.3.4 稳健的MPC

14.4 非线性分析和控制

14.5 领域随机化

14.6 延伸部分

14.6.1 替代的风险/稳健性衡量标准

参考文献

1. Kemin Zhou and John C. Doyle, "Essentials of Robust Control", Prentice Hall. 1997.

2. I.I. R. Petersen and V. A. Ugrinovskii and A. V. Savkin, "Robust Control Design using H-inØnity Methods", Springer-Verlag , 2000.

3. StephenBoyd 和 LievenVandenberghe ，《凸优化》， 剑桥大学出版社。 2004.

4. Christian Ebenbauer and Tobias RaÆ and Frank Allgower, "Dissipation inequalities in systems theory:An introduction and recent results", *Invited Lectures of the International Congress on Industrial and Applied Mathematics2007* , pp.23-42, 2009.

5. Carsten Scherer和Siep Weiland， "线性{矩阵}的{不等式}。 中的{不等式}。{控制}",在线草案,pp. 293,2015.

6. D.Jacobson, "Optimal stochastic linear systems with exponential performance criteria and their relation to deterministic diÆerential games", *IEEE Transactions on Automatic Control*, vol.18, no.2, pp.124-131, apr, 1973.

7. S.Boyd and S.-J. Kim and L. Vandenberghe and A. Hassibi, "A Tutorial on Geometric Programming", *Optimization and Engineering*, vol. 8, no. 1, pp.67-127, 2007.

8. Peter Whittle, "Risk-sensitive optimal control",Wiley New York, vol. 20, 1990.

9. Dylan Foster和Alexander Rakhlin， "超越{UCB}。Optimal and EÆicient Contextual Bandits with Regression Oracles", *Proceedings of the 37th International Conference on Machine Learning*, vol. 119, pp.3199-3210, 13-18 Jul, 2020.

低动能机器人技术

走路、跑步、游泳、 飞行和操纵的算法

吕斯-特德雷克

最后修改 2021-6-13.

注意：这些是用于在麻省理工学院教授的课程的工作笔记。 它们将在2021年春季学期中被更新。 讲座视频可在YouTube上找到。

上一章 目录 下一章

章节 15

输出反馈 （又称 "像素到扭矩"）。

在本章中，我们将开始考虑形式为的系统。

$$\mathbf{x}[n+1] = \mathbf{f}(\mathbf{x}[n], \mathbf{u}[n], \mathbf{w}[n], n)$$
$$\mathbf{y}[n] = \mathbf{g}(\mathbf{x}[n], \mathbf{u}[n], \mathbf{v}[n], n)。$$

换句话说，我们将最终开始解决我们必须根据传感器的测量结果做出决定的事实--到目前为止，我们的大部分讨论都默认了我们可以获得系统的真实状态，用于我们的反馈控制器（而这已经是一个难题了）。

在某些情况下，我们将看到，"全状态反馈 "的假设并不那么糟糕
-- 我们确实有很好的工具可以从原始传感器数据中进行状态估计。但是，即使是我们最好的状态估计算法也会给系统添加一些动态因素，以便剔除噪声测量；如果这些Ølters的时间常数接近我们动态因素的时间常数，那么我们在分析闭环系统时就必须包括估计器的动态因素。

在其他情况下，假设我们对系统的全部状态有一个估计，来设计一个控制器是完全过于乐观的。一些状态变量可能是完全不可观察的，其他的可能需要控制器采取特殊的"信息收集 "行动。

对我来说，机器人操纵的问题是一个应用领域，在这个领域，更直接的输出反馈方法变得至关重要。想象一下，你正试图为一个需要为你的礼服衬衫扣上纽扣的机器人设计一个控制器。如果第一步是估计衬衫的状态（我的衬衫有多少个自由度？），那么感觉我们就不会成功。或者，如果你想给机器人编程做沙拉 -- 沙拉的状态是什么？我真的需要知道每块生菜的位置和速度才能成功吗？

15.1 古典观点

在某种程度上，这种将 "输出反馈 "作为一个特殊的、高级的话题来称呼的想法是一个新问题。在状态空间和基于优化的控制方法迎来 "现代控制 "之前，我们有 "经典控制"。经典控制主要（但不完全）关注线性时变（LTI）系统，并大量使用频域分析（例如，通过傅里叶变换/拉普拉斯变换）。有许多关于这个问题的优秀书籍；[.12]是现代处理的很好的例子，它们从状态空间表示开始，但也处理频域的观点。

我们在这里要承认的是，在经典控制中，基本上所有的东西都是围绕着输出反馈的概念建立的。其基本概念是系统的传递函数，它是一个输入到输出的图谱（在频域），可以完全描述一个LTI系统。像磁极安置和环路整形这样的核心概念从根本上解决了我们在这里讨论的输出反馈的挑战。 有时我觉得，尽管我们在现代的、基于优化的控制方面获得了很多东西，但我担心我们在考虑闭环性能的丰富特征（上升时间、停留时间、超调.....）方面失去了一些东西，甚至可能在我们的系统对未建模的误差的实际鲁棒性方面失去了一些东西。

15.2 基于观察者的反馈

15.2.1 卢恩贝格观察家

15.2.2 带高斯噪声的线性二次调节器(LQG)

15.2.3 部分可观察的马尔科夫决策过程

15.3 静态输出反馈

15.3.1 对于线性系统

15.4 基于干扰的反馈

15.4.1 系统级合成

参考文献

1. Joao P. Hespanha，"线性系统理论"，普林斯顿出版社。 2009.
2. Karl Johan {AA}str{"o}m和Richard M Murray，"反馈系统：科学家和工程师的介绍"，普林斯顿大学出版社。 2010.

低动能机器人技术

走路、跑步、游泳、 飞行和操纵的算法

吕斯-特德雷克

最后修改 2021-6-13.
如何引用这些笔记，使用注释，并给予反馈。

注意：这些是用于在麻省理工学院教授的课程的工作笔记。它们将在2021年春季学期中被更新。讲座视频可在YouTube上找到。

上一章 目录 下一章

章节 16

在Colab中打开

极限循环的算法

章中关于行走和跑步机器人的讨论激发4了极限循环稳定性的概念。线性系统不能够产生稳定的极限循环行为，所以这个丰富的主题是非线性系统设计和分析所特有的。此外，设计、稳定和验证极限循环所需的工具将具有超越简单周期运动的适用性。

我们必须问的第一个自然问题是，给定一个系统$\dot{\mathbf{x}} = f(\mathbf{x})$，或一个控制系统$\dot{x} = f(\mathbf{x}, \mathbf{u})$，我们如何去寻找可能是被动稳定、开环稳定或通过闭环反馈稳定的周期性解决方案？事实证明，我们已经开发的轨迹优化工具非常适用于这项任务。

16.1 轨迹优化

我介绍了轨迹优化工具，作为从一个特定的已知初始条件开始优化控制轨迹的手段。但是，对系统的各个轨迹进行优化的基本思想在更大范围内是有用的。即使对于一个被动系统，我们也可以把对周期性解决方案的搜索表述为对满足动态约束和*周期性约束的*轨迹的优化，$\mathbf{x}[0] = \mathbf{x}[N]$。

$$\begin{array}{ll} \text{发现} & \\ \mathbf{x}[\cdot] & \text{受制于} \mathbf{x}[n+1] = f(\mathbf{x}[n])。 \qquad \forall n \in [0, N-1] \\ & \mathbf{x}[0] = \mathbf{x}[N]。 \end{array}$$

当然，我们也可以将控制输入添加到配方中，但让我们从这个简单的案例开始。花点时间思考一下这个问题表述的可行方案。当然，一个Øxed点$\mathbf{x}[n] = \mathbf{x}^*$将满足约束条件；如果
我们不希望这些解从求解器中出来，我们可能需要用约束条件来排除它们，或者添加一个目标来引导求解器走向所需的解。其他可能的解决方案是恰好在N步内周期性的轨迹。这是很有限制性的。

如果我们使用连续时间的公式，我们可以做得更好。例如，在我们的

引入直接搭配，我们写道

$$\begin{aligned} &\text{闵}_{\mathbf{x}[\cdot],\mathbf{u}[\cdot]} && \ell_f(\mathbf{x}[N]) + \sum_{n_0}^{N-1} h_n \ell(\mathbf{x}[n], \mathbf{u}[n]) \\ &\text{受制于} && \dot{\mathbf{x}}(t_{c,n}) = f(\mathbf{x}(t_{c,n}), \mathbf{u}(t_{c,n}))。 \qquad \forall n \in [0, \ N-]。1 \\ & && \mathbf{x}[0] = \mathbf{x}_0 \\ & && + \text{额外的限制。} \end{aligned}$$

但是我们也可以在优化中加入h_n作为决策变量（提醒：我建议设置一个下限$h_n \geq h_{min} > 0$）。这使得我们的N步轨迹优化可以扩展和收缩时间，以满足周期性约束。
其结果是简单而有力的。

例子 (16.1寻找范德波尔振荡器的极限周期)

回顾一下范德波尔振荡器的动力学，由以下公式给出

$$\ddot{q} + \mu(q^{-2})\dot{q}1 + q = 0。 \quad \mu > 0,$$

其中表现出稳定的极限循环。形成直接配位优

化。

$$\begin{aligned} &\text{找到}_{\mathbf{x}[\cdot],h} && \text{遵循 } \mathbf{q}[0] = 0, && \dot{q}[0] > 0, \\ & && \mathbf{x}[N]=\mathbf{x}[0], && \text{（周期性约束）搭配动} \\ & && \text{态约束} \\ & && 0.01 \leq \mathsf{h} \leq 0.5 \end{aligned}$$

试试吧。

在Colab中打开

像往常一样，确保你看一下代码。摸索一下。尝试改变一些东西。

你应该注意到代码中的一件事是我为求解器提供了一个初始猜测。在迄今为止的大多数例子中，我都能够避免这样做--求解器将小的随机数作为默认的初始猜测，然后从那里求解。但是对于这个问题，我发现它被卡在了一个局部最小值中。添加初始猜测，即解在状态空间中绕圈移动就足够了。

16.2 利亚普诺夫分析

回顾一下，轨迹在时间上的稳定性和极限循环的稳定性之间的重要区别是，极限循环在相位上不收敛--靠近循环的轨迹向循环收敛，但循环上的轨迹可能不会相互收敛。
这是一种稳定性，也被称为*轨道稳定性，*可写为
作为对由轨迹$\mathbf{x}^*(t)$描述的流形的稳定性。

$$\min_{\tau} ||x(t) - x(^*\tau)|| \to 0.$$

在极限循环的情况下，这个流形是一个周期性的解决方案，具有
$\mathbf{x}^*(t+_{tperiod}) = \mathbf{x}^*(t)$。根据这种收敛的确切发生方式，我们可以
在Lyapunov、渐进轨道稳定性、指数轨道稳定性，甚至是Ønite-time轨道稳定性的意义上确定轨道稳定性。

为了证明一个系统是轨道稳定的（局部的，在一个区域内的，或全局的），或分析一个极限循环的吸引区域，我们可以使用李亚普诺夫函数。当我们分析步行和跑步的简单模型时，我们对Poincare图进行了分析。事实上，如果我们能找到一个李亚普诺夫函数，证明离散Poincare图的（i.s.L., asympototic, or exponential）稳定性，那么这就意味着（i.s.L, asymptotic, or exponential）周期的*轨道*稳定性。但是这种形式的李亚普诺夫函数是难以验证的，因为有一个相当根本的原因：我们很少有Poincare图的分析表达式，因为它是对周期的非线性动力学进行整合的结果。

相反，我们将把注意力集中在构建全状态空间的李亚普诺夫函数上，它使用连续动力学。特别是，我们希望考虑具有以下形式的李亚普诺夫函数；它们沿周期的任何地方都消失为零，而在远离周期的任何地方都是严格的正值。

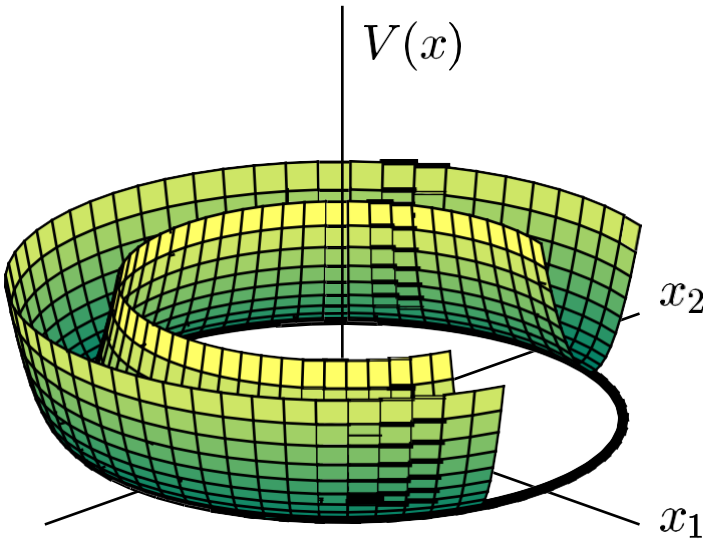


图-16.1-李亚普诺夫函数的漫画，该函数在极限周期上消失，其他地方都是严格的正值。(仅仅为了可视化的目的，删除了一小段）。

16.2.1 横向坐标

我们怎样才能将这一类函数参数化呢？对于任意的循环，这可能是非常diÆicult的原始坐标。对于像漫画中那样的简单周期，我们可以想象使用极坐标。更一般地说，我们将建立一个相对于轨道的新坐标系，其坐标为

- τ - 沿着轨道的相位
- $\mathbf{x}(_{\perp}\tau)$ - 剩余的坐标，与τ线性独立。

给定原坐标中的状态$\mathbf{x}$，我们必须为这个新的坐标系设计一个平滑的映射$\mathbf{x}\rightarrow$（τ，$\mathbf{x}_{\perp}$）。例如，对于一个简单的环形振荡器，我们可能有。

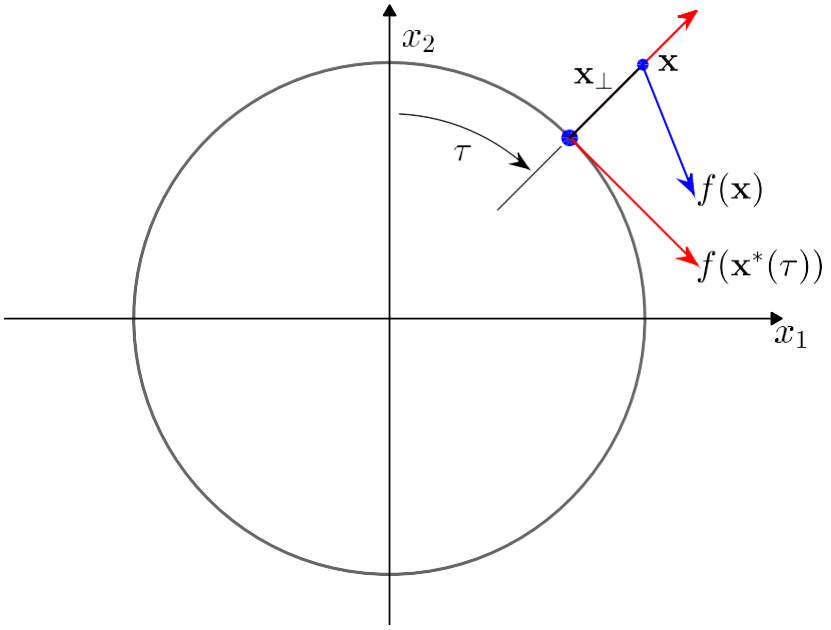


图-16.2-沿极限周期的移动坐标系。

一般来说，对于一个n维的状态空间，τ将总是一个标量，而$\mathbf{x}_\perp$将是一个$(n-1)$维的矢量，相对于$\mathbf{x}^*(\tau)$而言，将其余坐标去掉。

.事实上，尽管我们使用了$\mathbf{x}_\perp$这个符号，但坐标系不需要严格地与轨道正交，而必须是*横向的*（不是平行的）。在定义了平滑映射$\mathbf{x}\rightarrow(\tau,\ \mathbf{x}_\perp)$之后，我们总是可以在这个新的坐标系中重写动力学。

$$\dot{\tau} = f_\tau(\mathbf{x}_\perp, \tau)$$
$$\dot{\mathbf{x}}_\perp = f_\perp(\mathbf{x}_\perp, \tau)。$$

为了在这些说明的其余部分保持我们的符号简单，我们将假设这个新坐标系的原点在名义轨迹上（$\min_\tau |\mathbf{x}-\mathbf{x}^*(\tau)| = \Rightarrow 0\ \mathbf{x}_\perp = 0$）。同样地，通过将$\tau$作为相位变量，我们将利用这一事实，在名义轨迹上，我们有$\dot{\tau} = f_\tau(0,\tau) = 1$.

这种结构对李亚普诺夫分析的价值是在[1]中提出，并在[]中被很好地扩展到控制设计中。2]中对控制设计进行了很好的扩展，并在[]中对吸引力区域的估计进行了扩展。3].在[ ]中给出了一个相当普遍的用于确定横向坐标的数值策略。4].

定理-16.1-轨道稳定性的李亚普诺夫定理

对于一个动态系统$\dot{\mathbf{x}}=f(\mathbf{x})$，$\mathbf{x}\in R^n$，$f$连续，一个连续的周期性解$\mathbf{x}^*(\tau)$，以及一个光滑的映射$\mathbf{x}\rightarrow(\tau,\ \mathbf{x}_\perp)$，其中$\mathbf{x}$在$\mathbf{x}^*$上$_\perp$消失，那么对于原点周围的一些$n1$维球B，如果我可以产生一个$V(\mathbf{x}_\perp,\ \tau)$，使得

$$\forall\tau,\ V(0,\tau)=0\ ,$$
$$\forall\tau,\quad \forall\mathbf{x}_\perp\in \mathrm{B},\quad \mathbf{x}_\perp\neq 0,\quad V(\mathbf{x}_\perp,\ \tau)>0。$$

与

$$\forall\tau,\ \dot{V}(0,\tau)=0,$$
$$\forall\tau,\quad \forall\mathbf{x}_\perp\in \mathrm{B},\quad \mathbf{x}_\perp\neq 0,\quad \dot{V}(\mathbf{x}_\perp,\ \tau)<0。$$

则解$\mathbf{x}^*(t)$是*局部轨道渐近稳定的*。

李亚普诺夫意义上的轨道稳定性和指数轨道稳定性也可以得到验证（分别为$\dot{V}_{\perp} \leq$和$0\dot{V}_{\perp} \leq \alpha_{\perp} V$）。

示例（16.2简单的环形振荡器）

也许最简单的振荡器是收敛于单位圆的Ørst-order系统。在笛卡尔坐标中，其动力学特性为

$$\dot{x}_1 = x_2 - \alpha x_1 \left(1 - \frac{1}{\sqrt{x_1^2 + x_2^2}}\right)$$

$$\dot{x}_2 = -x_1 - \alpha_2 \left(1x - \frac{1}{\sqrt{x_1^2 + x_2^2}}\right),$$

其中α是一个正的标量增益。

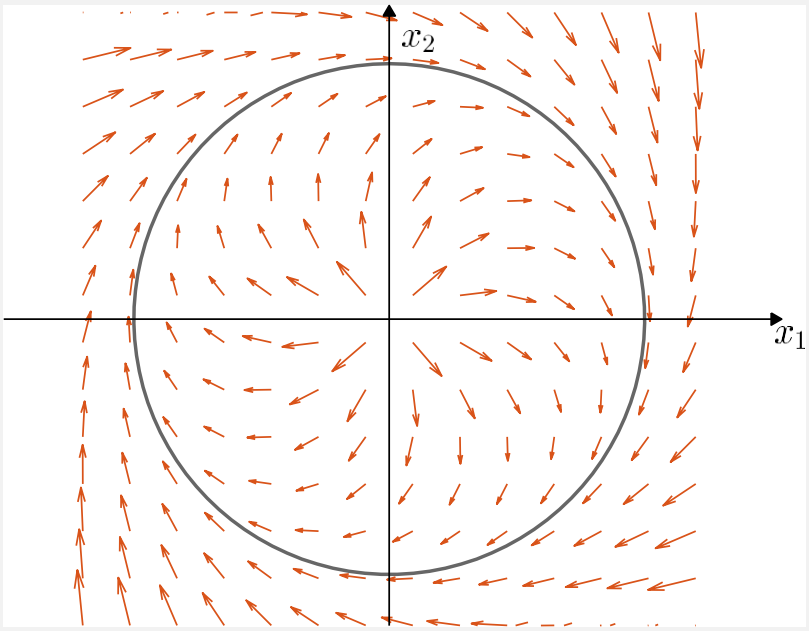


图 -16.3 环形振荡器的矢量Øeld

如果我们把横向坐标看作是极坐标，移位后，使
$x_{\perp}$在单位圆上为零。

$$\tau = \operatorname{atan2}(-x_2, x_1)$$
$$x_{\perp} = x\sqrt{x_1^2 - 1, 2}$$

这在$x_{\perp} > -$时1有效，那么简单的横向动力学就被揭示出来。

$$\dot{\tau} = 1$$
$$\dot{x}_{\perp} = -\alpha_{\perp} x$$

以Lyapunov候选者$V(x_{\perp}, \tau) = x$为例²，我们可以验证以下几点

$$\dot{V} = -\alpha 2x \leq 20, \quad \forall_{\perp} x > -1.$$

这证明了极限循环是局部渐进稳定的，此外，不变的开放集V<1位于该极限循环的吸引区域内。事实上，我们知道所有的$x_{\perp}$>-1都在该极限循环的吸引区域内，但这并没有被上述的李亚普诺夫论证所证明。

让我们再次将这种方法与我们在步行机器人一章中使用的方法进行比较，在该章中我们使用了Poincaré地图分析来研究极限循环稳定性。在横向坐标方法中，有一个额外的负担，即沿着整个轨迹构建坐标系，而不是只在单一的断面上。事实上，横向坐标法有时被称为 "移动的Poincaré截面"。同样，这个额外工作的回报是，我们可以检查一个只涉及瞬时动态的条件，即$f(\mathbf{x})$。
而不是在整个周期中整合动力学，以产生离散的Poincaré图，$\mathbf{x}_p[n+1]=P(\mathbf{x}_p[n])$。正如我们将在下面看到的，这种方法也将与设计稳定极限循环的连续反馈控制器更加兼容。

16.2.2 横向线性化

在围绕Øxed点的李亚普诺夫分析中，有一个重要的特例：对于稳定的线性系统，我们实际上有一个构造李亚普诺夫函数的配方。因此，对于非线性系统，我们经常发现，通过围绕Øxed点线性化并使用线性系统的李亚普诺夫候选函数作为初始猜测来开始我们的搜索是很方便的。这种方法也可以扩展到极限循环分析。

特别是，让我们对横向动力学做一个线性近似。

$$\dot{\mathbf{x}}_\perp = f_\perp(\mathbf{x}_\perp, \tau) \approx \frac{\partial f_\perp}{\partial \mathbf{x}_\perp}\mathbf{x}_\perp = \mathbf{A}_\perp(\tau)\mathbf{x}_\perp.$$

请注意，我已经假定$\mathbf{x}_\perp$在标称轨迹上为零，所以不需要在这里用误差坐标工作的额外符号。请记住，τ是沿轨迹的相位，但是[1]表明，*时间变化的*线性系统$\dot{\mathbf{x}}_\perp = \mathbf{A}_\perp(t)\mathbf{x}$的（指数）稳定性$_\perp$意味着原始系统的局部指数轨道稳定性。特别是，如果这个横向线性系统是周期性的和轨道稳定的，那么对于每个$\mathbf{Q} = {}^{QT}\,00$，存在一个唯一的周期性Riccati方程的正DeØnite解。

$$V(\mathbf{x}_\perp, \tau) = \mathbf{x}^T\mathbf{P}(\tau)\mathbf{x}, \qquad {}_\perp\mathbf{P}^T(\tau) = \mathbf{P}(\tau), \quad \mathbf{P}(\tau + t_{period}) = \mathbf{P}(\tau)。$$
$$-\dot{\mathbf{P}}(\tau) = \mathbf{P}(\tau)\mathbf{A}(\tau) + \mathbf{A}(\tau)\mathbf{P}^T(\tau) + \mathbf{Q}, \qquad \mathbf{Q} = \mathbf{Q}^T 0\ .0$$

关于寻找这些Lyapunov（和相应的Riccati）方程的周期性解的数值方法，有很多令人惊讶的文献。在实践中，对于我曾经尝试过的每一个问题，我都是简单地将方程向后积分，直到积分收敛到一个周期解（这并不保证有效，但几乎总是有效）。

这是很强大的。它给我们提供了一个一般的方法来构建一个Lyapunov候选人，以证明循环的局部稳定性，并可作为非线性Lyapunov分析的候选人。

16.2.3 使用平方之和的吸引力区域估计

即将推出。如果你有兴趣，请看 [43].

16.3 反馈设计

我们为稳定Øxed点或稳定轨迹所开发的许多工具都可以适应这种新的设置。 当我们讨论弹簧加载的倒立摆模型的控制时，我们考虑了离散的控制决策，每周期一次，这可以稳定离散的Poincare图，$\mathbf{x}_p[n+1]=f_p(\mathbf{x}_p[n], \mathbf{u}[n])$， 其中$\mathbf{u}[n]$是每周期一次的决策。虽然'它有可能

使用我们的邻接方法得到这个地图的局部线性近似，记住我们很少有非线性f的$_p$分析表达。

将我们自己限制在每周期一次的决策中，这有多大的局限性？当然，我们可以通过持续的反馈来改善一般的性能。但在某些情况下，每周期一次的决定也是很合理的。腿部机器人的放脚是一个很好的例子--一旦我们选择了放脚的位置，我们可能会在放脚之前做一些小的修正，但一旦脚踏上地面，就很少改变这个决定；每步一次似乎是一个合理的近似值。另一个很好的例子是∞拍翼飞行：在这里，拍翼的周期时间可能比质心的动态时间尺度快得多，而改变拍翼行程的参数，每次∞拍一次似乎很合理。

但是我们上面开发的工具也给我们提供了考虑整个轨迹的连续反馈所需的机制。让我们来看看几个重要的公式。

16.3.1 对于行动不足的程度一。

事实证明，我们的许多简单的行走模型--特别是那些用最小坐标推导出的点状脚--只缺少一个执行器（在脚和地面之间）。我们甚至可以用这种方式来表示相当复杂的机器人；我在这里要回避的许多理论最初是由Jessy Grizzle开发的
他和他的小组在双足机器人RABBIT的背景下进行了研究。杰西的主要观点是，极限循环稳定性实际上是n个1自由度的稳定性，而且你通常可以通过n个1执行器轻松实现它--他将这一工作路线称为 "混合零动力学"（HZD）。我们将在下一章中讨论 "混合 "的部分。
章，但这里有一个简单的例子来说明 "零动力 "的概念。[即将推出...]

稳定零点动力学是实现轨道稳定的同义词。有趣的是，它实际上并不能保证你会绕过极限周期。HZD步行者的一个 "特点 "是，你实际上可以站在它们前面，用你的手停止它们的前进 -- 控制系统将继续行动，在周期上保持自己，但沿着周期的进展已经停止。在一些机器人中，你甚至可以将它们向后推，它们将以反向方式完成同样的行走周期。如果人们想证明将实现向前的进展，那么任务就减少到检查沿周期的相位变量的一维动态。

"零动力学 "的概念当然不限于具有一度欠激励的系统。一般来说，我们可以很容易地用m个执行器稳定一个维度为m的流形（我们在任务空间部分反馈线性化一节中看到了这一点），如果在该流形上是足够实现我们的任务的，或者如果我们可以证明流形上产生的动力学对我们的任务是足够的，那么生活是美好的。但在 "欠作用度一 "的情况下，所研究的流形是一个轨迹/轨道，本章的工具可以立即适用。

16.3.2 横向LQR

稳定循环的另一个自然方法是使用横向线性化。特别是，对于一个名义（控制）轨迹$[\mathbf{x}_0(t), \mathbf{u}_0(t)]$，$\dot{\mathbf{x}}_0 = f(\mathbf{x}_0, \mathbf{u}_0)$，我们可以对横向动力学进行近似。

$$\dot{\mathbf{x}}_\perp = f_\perp(\mathbf{x}_\perp, \tau, \mathbf{u}) \approx \frac{\partial f_\perp}{\partial \mathbf{x}_\perp}\mathbf{x}_\perp + \frac{\partial f_\perp}{\partial \mathbf{u}}(\mathbf{u} - \mathbf{u}_0(\tau)) = \mathbf{A}_\perp(\tau)\mathbf{x}_\perp + \mathbf{B}_\perp(\tau)\bar{\mathbf{u}}。$$

[2] 表明稳定该系统的（周期性）时变LQR控制器实现了原系统的*轨道*稳定。

让我们花点时间来体会这种方法与时间的区别。

在全坐标上变化的LQR。当然，行为是完全不同的：全坐标上的时变LQR将尝试减速或加速，以保持与标称轨迹的时间一致，而这个控制器没有尝试稳定相变量。你可能认为你可以设计原始的时变控制器，$\bar{\mathbf{u}} = \mathbf{K}(t)\bar{\mathbf{x}}$，在执行过程中只需 "投射到最接近的时间"（例如Ønd the time $t = \text{argmin}_\tau |\mathbf{x} - \mathbf{x}(_0\tau)|$）并使用与该时间对应的反馈。但是，这种投射并不能保证安全；你可以发现通过横向线性化可以稳定的系统在使用基于这种投射的反馈时在闭环中是不稳定的。

例子（16.3的变的LQR不是作时间上的投影会导致不稳定）

但还有一个更重要的原因，就是喜欢横向LQR方法。如果轨道稳定性是你所需要的全部，那么这是一个更好的问题表述，因为你要求的是更少。正如我们刚刚讨论的混合零动力学，用m个执行器来稳定m个自由度与稳定$m+1$个自由度可能有很大不同。在横向公式中，我们要求LQR仅在横向坐标中最小化成本（这在数学上相当于在全坐标中设计推出的成本函数，在标称轨迹方向上为零）。在实践中，这可能导致更小的成本-去向矩阵$\mathbf{S}(t)$和更小的反馈增益$\mathbf{K}(t)$。对于欠激励的系统。

这种差异可能是戏剧性的。

16.3.3 非周期性轨迹的轨道稳定化

观察到轨迹的轨道稳定可以对你的欠激励控制系统提出更少的要求（并导致更小的LQR反馈增益），这是相当强大的。它并不限于稳定周期性运动。如果你的目的是稳定一个通过状态空间的非周期性路径，但实际上并不关心时间，那么制定稳定化可以是一个非常好的选择。你可以找到这样的例子：一个系统在横坐标上是可稳定的，但在全坐标上却不是。

例子 16.4（在全坐标中是稳定的，但在横坐标上是稳定，但在全坐标上不是）

不过要小心，因为反过来也可能是真的：一个系统有可能在全坐标中是稳定的，但在某些横坐标中，如果你对这些坐标选择得不好的话。

例16.5（在全坐标中可稳定，但在横坐标中不可稳定）。

设计横向坐标的一般方法[]可以解决这个问题。

考虑一个$u(t)=1$（来自任何初始条件）的双积分器的轨迹。我们当然可以用LQR将这个轨迹稳定在全坐标上。从技术上讲，我们可以选择横向坐标$\tau = q$，$x_\perp = q$。$x_\perp$在任何地方都是横向于名义轨迹的。 但这种横向的线性化也显然是不稳定的。这是个糟糕的选择!

通过在优化横向坐标时加入可控性标准来解决这一问题。

参考文献

1. John Hauser和Chung Choo Chung, "Converse {L}yapunov functions for exponentially stable periodic orbits", *Systems & Control Letters*, vol. no23,. pp. 1,1994.

2. Anton S. Shiriaev and Leonid B. Freidovich and Ian R. Manchester, "Can We Make a Robot Ballerina Perform a Pirouette?欠动力机械系统周期性运动的轨道稳定", 《*控制学年刊*》, 第32,200-211页2,, 12月。 2008.

3. Ian R. Manchester and Mark M. Tobenkin and Michael Levashov and Russ Tedrake, "Regions of Attraction for Hybrid Limit Cycles of Walking Robots", *Proceedings of 18th IFAC World Congress, extended version available online: arXiv: 1010.2247 [math.OC],* 2011.

4. Ian R. Manchester, "Transverse Dynamics and Regions of Stability for Nonlinear Hybrid Limit Cycles", *Proceedings of 18th IFAC World Congress, extended version available online: arXiv:1010.2241 [math.OC]*, Aug- Sep, 2011.

低动能机器人技术

走路、跑步、游泳、 飞行和操纵的算法

吕斯-特德雷克

最后修改 2021-6-13.
如何引用这些笔记，使用注释，并给予反馈。

注意：这些是用于在麻省理工学院教授的课程的工作笔记。它们将在2021年春季学期中被更新。讲座视频可在YouTube上找到。

上一章 目录 下一章

章节 17

规划 和 通过联系的 Coolabl

到目前为止，我们已经开发了一个相当强大的工具箱，用于规划和控制 "平滑 "系统--系统的运动方程由一个到处都是平滑的函数$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u})$描述。 但是，我们对腿部机器人的简单模型的讨论表明，与世界产生和中断接触的动力学更为复杂--这些通常被建模为混合动力学，在碰撞事件中具有冲击不连续性和受限的
接触过程中的动态变化（有软约束或硬约束）。

我在这一章的目标是将我们的计算工具扩展到这一类更丰富的模型。我们的许多核心工具仍然有效：轨迹优化、李亚普诺夫分析（例如用平方之和）和LQR都有自然的对应物。

让我们从一个热身练习开始：无边轮的轨迹优化。我们已经有了这方面所需的基本内容，它将为我们在整个章节中推广我们的方法打下良好基础。

例子（17.1无边轮的轨迹选择）。

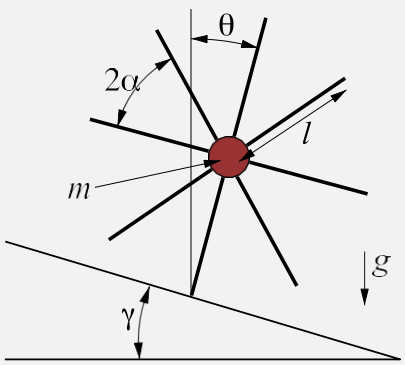


图-17.1-无边轮。姿态腿的方向，ϑ，从垂直轴顺时针测量。

无边轮是我们最简单的被动-动态步行器的例子；它没有控制输入，但表现出被动的稳定滚动Øxed point。我们也已经看到，轨迹优化可以作为一个工具，用于Ønding平滑无源系统的极限循环，例如通过制定一个直接拼合问题。

$$\begin{aligned} \underset{\mathbf{x}[\cdot],h}{\text{发现}} \quad & \text{受制于拼接约束} (\mathbf{x}[n], \mathbf{x}[n+1], h), && \forall n \in [0, N-1] \\ & \mathbf{x}[0] = \mathbf{x}[N]。 \\ & h_{min} \le h \le h_{max}。 \end{aligned}$$

其中h是轨迹断点之间的时间步长。

事实证明，将其应用于无边轮是非常直接的。我们仍然希望找到一个周期性的轨迹，但现在必须考虑到碰撞事件。我们可以通过修改周期性条件来做到这一点。让我们强制初始状态为碰撞后，最终状态为碰撞前，并确保它们通过碰撞方程相互关联。

$$\begin{aligned} \underset{\mathbf{x}[\cdot],h}{\text{发现}} \quad & \text{受制于拼接约束} (\mathbf{x}[n], \mathbf{x}[n+1], h), && \forall n \in [0, N-1] \\ & \vartheta[0] = \gamma - \alpha。 \\ & \vartheta[N] = \gamma + \alpha。 \\ & \dot{\vartheta}[0] = \dot{\vartheta}[N] \cos(2\alpha) \\ & h_{min} \le h \le h_{max}。 \end{aligned}$$

虽然这个简单的例子可能不需要（因为动力学是有限的），但为了完整起见，我们还应该增加约束条件，以确保没有一个中间点是接触的。

$$\gamma - \alpha \le \vartheta[n] < \gamma + \alpha, \quad \forall n \in [1, N-1]。$$

其结果是一个简单而干净的数值算法，用于确定无轮辋的滚动极限周期解决方案。

请试一试吧。

在Colab中打开

无边轮的具体情况是相当干净的。但在我们把它应用于罗盘步态、膝行罗盘步态、弹簧式倒立摆等之前，我们应该停下来，找出一个更普遍的形式。

17.1 (自主) 混合系统

回顾一下我们是如何建立简单腿部机器人的动力学模型的。首先，我们为每一种可能的接触配置推导出运动方程（独立的）--例如，在弹簧加载的倒立摆（SLIP）模型中，我们有一套方程来控制质量在飞行阶段的（x，y）位置，还有一套完全独立的方程，用极坐标（r，ϑ）来描述站立阶段的。然后，我们做了一些额外的工作来描述这些模型之间的过渡--例如，在SLIP中，当脚第一次接触地面时，我们从飞行阶段过渡到站立阶段。当模拟这个模型时，这意味着我们有一个离散的 "事件"，它发生在脚碰撞的瞬间，并且机器人的状态立即发生了不连续的变化（在这种情况下，我们甚至改变了状态变量）。

*混合系统*的语言为我们提供了描述这种形式的系统的丰富语言，以及一套分析和控制它们的工具。术语 "混合系统 "有点超载，在这里我们用 "混合 "来表示离散和

我们在此考虑的特定系统有时被称为*自主*混合系统，因为内部动力学可以在没有任何外在输入的情况下引起离散变化。在混合系统的表述中，我们用一组*模式来*描述一个系统，每个*模式*都由（理想的平滑）连续的

动态，一组*守卫*，在这里是连续的函数，其零级
里是连续的。
描述了触发事件的条件，以及一组*复位*，描述了由防护措施触发的对状态的离散更新。每个卫兵都与一个特定的模式相关联，我们可以在每个模式下有多个卫兵。每个守卫最多只有一个复位。你偶尔会在这里把守卫称为 "见证函数"，因为它们在模拟中扮演着这个角色，而复位有时被称为 "过渡函数"。

与此相对应的是，比如说："我们的模式"。
集的动力火车，在这

变速器是一种外部输入。

我喜欢把混合系统的想象力保留在脑海中，下面是一个简单的例子，即机器人的脚跟撞击地面。混合系统的一个解决方案轨迹在每个模式内都有一个连续的轨迹，当轨迹碰到守卫的零级集（这里脚跟和地面之间的距离变为零）时，由离散的更新来点缀，重置描述了状态变量的离散变化。

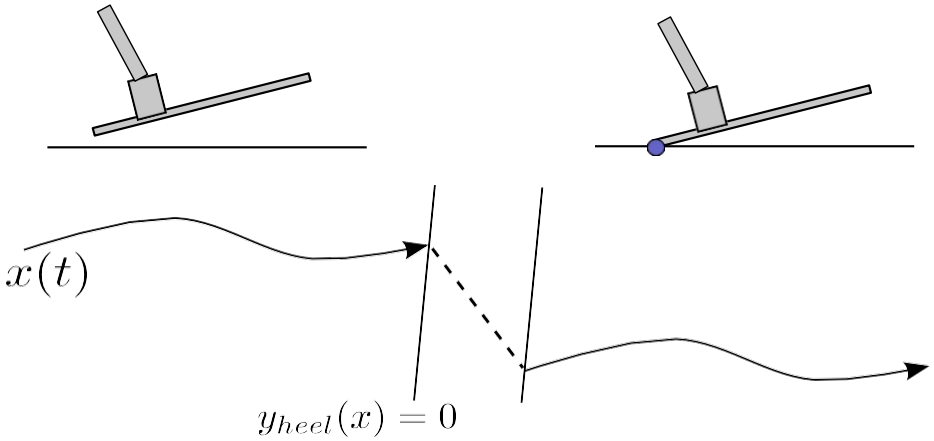


图 -17.2 将接触作为一个混合系统进行建模。

对于这种机器人的脚，我们可以把动力学分解成不同的模式。(1)脚在空中，(2)只有脚跟在地上，(3)脚跟和脚趾在地上，(4)只有脚趾在地上（推-oÆ）。更一般地说，我们将模式*i*的动力学写成$\mathbf{f}_i$，将模式*i*过渡到模式*j*的信号警卫写成$j_{i,j}$（其中模式*i*内$j_{i,j}(\mathbf{x}_i)>0$），将*i*到*j*的复位图写成$\mathbf{\Delta}_{i,j}$，如下图所示。

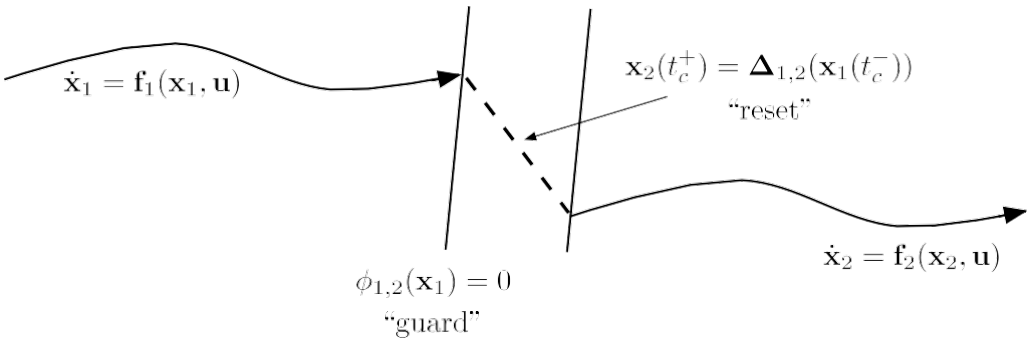


图-17.3-混合系统的语言：模式、守卫和重置图。

17.1.1 混合轨迹优化

使用模式、守卫和重置的更一般的语言，我们可以开始制定 "混合轨迹优化 "问题。

在混合轨迹优化中，有一个重要的区别，即轨迹优化中的*模式序列是预先知道的*，优化只是试图解决

的连续时间轨迹，与此同时，我们还必须发现模式序列。

对于模式序列是Øxed的情况，那么混合轨迹优化就像把多个单独的数学程序拼接成一个数学程序一样简单，边界条件的约束是为了执行防护/复位约束。对于一个简单的混合系统，有一个给定的模式序列，并使用短语$\mathbf{x}_k$等表示序列中第k段的模式状态，我们可以写出。

发现 $x_k[\cdot], h_k$ 受制于$\mathbf{x}[0_0]=\mathbf{x}_0$。

$$\forall k\ j_{k,k+1}(\mathbf{x}_k[N_k]) = 0,$$
$$\mathbf{x}[_{k+1}0] = \mathbf{\Delta}(_{i,j}\mathbf{x}_k[N_k])。$$
$$\phi_{k,k'}(\mathbf{x}_k[n_k]) > 0\ ,\ \forall n_k \in [0, N]_k, \forall k'$$
$$h_{min} \leq h_k \leq h_{max}。$$
搭配约束$_k$（$\mathbf{x}[_k n_k], \mathbf{x}_k[n_{\ k}+1], h_k$）。 $\forall n_k \in [0,\ N_k-]1。$

然后很自然地增加控制输入（作为额外的决策变量），并增加一个目标和任何更多的约束。

例子 (17.2一个篮球的技巧性投篮。)

作为这种混合轨迹优化的一个简单例子，我想看看我们是否能制定出优化篮球 "技巧性投篮 "的初始条件的搜索，会很有趣。通过快速搜索，我找到了这个视频的灵感。

让我们从更简单的开始 -- 只用一个 "弹跳传球"。我们可以用一些非常简单的动力学来捕捉一个弹跳球的动态（在平面内，忽略旋转）。

$$\mathbf{q}=[^{x}_{z}]。 \qquad \ddot{\mathbf{q}} = [\begin{matrix}0\\-g\end{matrix}]。$$

在任何没有接触的持续时间h的时间间隔内，我们实际上可以完美地整合这些动力学。

$$\mathbf{x}(t+h) = \begin{bmatrix} x(t) + h\dot{x}(t) \\ z(t) + h\dot{z}(t) - \frac{1}{2}gh^2 \\ \dot{x}(t) \\ \dot{z}(t) - hg \end{bmatrix}。$$

对于反弹通道，我们只考虑与地面的碰撞，所以我们有一个防护，$j(\mathbf{x}) = z$，当$z = 0$时触发，还有一个复位图，假设弹性碰撞的恢复系数e。

$$\mathbf{x}^{+} = \mathbf{\Delta}(\mathbf{x}^{-}) = [x^{-}\ z^{-}\ \ \dot{x}^{-}\ -e\dot{z}^{-}]^{T}。$$

我们将问题表述为：给定一个初始球位（x=0，z=1），一个4米外的Ønal球位（x=z4,=1），找出实现该目标的初始速度（秒5）。显然，这意味着$\dot{x}(0) = 4/5$。有趣的问题是--我们应该如何处理$\dot{z}(0)$？有多种解决方案--涉及不同次数的弹跳。我们可以通过一个简单的混合轨迹优化来找到所有的解决方案，利用这样的观察，即每个弹跳次数都有两种可能的解决方案--一种是以正的$\dot{z}(0)$开始，一种是以负的$\dot{z}(0)$开始。

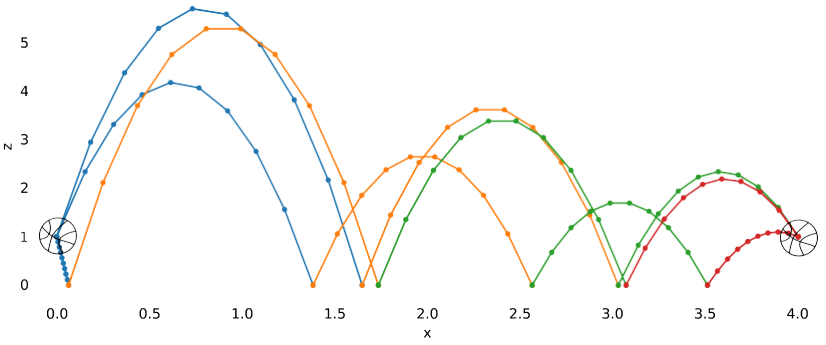


图17.4 - 轨迹优化到Ønd "弹跳传球 "的解决方案。我绘制了所有为或2,3,4弹跳找到的解决方案.....但我认为如果你在球场上使用这个，最好坚持使用单一的弹跳。

现在让我们来试试我们的技巧射击。我将把我们的目标移到x_f=−1m，z_f=3m，并在x=0处引入一堵垂直墙，然后把我们的初始条件移回x_0=−.25m。碰撞动力学，现在必须考虑到球的旋转，在附录中。第一次反弹是针对墙，第二次是针对∞门。
我还将约束Ønal速度向下（必须从上面接近环形物）。试试吧。

图-17.5-"巧射 "的轨迹优化。除了网底什么都没有!观众们都很兴奋!

在这个例子中，我们可以用分析的方法整合每个片段的动态。这只是例外，而不是规则。但你可以采取同样的步骤，多用一点代码，例如直接转录或在每个段中有多个断点的搭配。

17.1.2 稳定的混合模型。

17.1.3 推导出混合模型：最小坐标与Poating-base坐标

为了推导出这种形式的运动方程，还有一些工作要做。你还记得我们是如何做无边轮和罗盘步态的例子的吗？在这两种情况下，我们都假设有一只脚是固定在地面上的，而且它不会打滑，这使得我们可以把拉格朗日写成有一个针的样子

我们将脚与地面相连的关节的运动方程计算出来。对于SLIP模型，我们使用不同的拉格朗日方程来推导∞飞行阶段和站立阶段，每个方程都有不同的状态表示。我将此描述为*最小坐标*建模方法--它很优雅，并且具有一些重要的计算优势，我们将在下面的算法中体会到这些优势。但这是一个很大的工作!例如，如果我们还想考虑无边轮的脚部接触的摩擦力，我们将不得不推导出另一组方程来描述滑动模式（例如，增加一个棱形关节，使脚沿着斜坡移动），再加上计算给定状态下的接触力以及与摩擦锥边界的距离的守卫，等等。

幸运的是，有一种替代的建模方法可以推导出接触的模式、防护和复位，这种方法更为普遍（尽管承认也更为复杂）。我们可以用*Woating-base坐标对*机器人进行建模--我们添加一个Øctitious六自由度的"∞oating-base "关节，将机器人的某些部分与世界相连（在平面模型中，我们只使用三个自由度，例如（x, z, ϑ））。我们可以一次性推导出 "浮动基 "机器人的运动方程。
在不考虑接触的情况下，再加上因接触而产生的额外约束，作为接触力施加在身体上。由此产生的操纵器方程的形式为

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} = \tau_g(\mathbf{q}) + \mathbf{B}\mathbf{u} + \sum_i \mathbf{J}_i^T(\mathbf{q})\lambda_i, \tag{1}$$

其中λ_i是约束力，$\mathbf{J}_i$是约束雅各布。方便的是，如果我们的接触方程中的守护函数是有符号的接触距离$j_i(\mathbf{q})$，那么这个Jacobian就是简单的$\mathbf{J}_i(\mathbf{q}) = \frac{\partial \phi_i}{\partial \mathbf{q}}$。我在附录中写了常见情况（位置约束、速度约束、冲击方程等）的基本推导。这里需要理解的是，这是一个替代性的
如果我们用三个构型变量来写无边轮的方程，那么它就不再是一个最小的坐标系--运动方程是用$2N$个状态变量来写的，但系统实际上可能被限制为只能沿着一个较低维度的流形演化（如果我们用三个构型变量来写∞oating base的无边轮方程，当它处于姿态和摩擦锥内时，它仍然只能围绕脚尖旋转）。这将对我们的算法产生影响。

17.2 练习题

练习（17.1寻找罗盘步态极限周期）

在这个练习中，我们使用轨迹优化来确定指南针步态的极限周期。我们使用了一种相当普遍的方法：机器人的动力学是用∞oating-base坐标来描述的，摩擦接触也被准确地建模了。在这个笔记本中，你被要求对这个优化问题所需要的许多约束进行编码。

a. 在所有的突破点上强制执行站立脚和地面之间的接触。
b. 在最初的时候，强制执行摆动脚和地面之间的接触。
c. 防止摆动脚在所有的突破点上穿透地面。(在这个分析中，我们将忽略摆动脚和地面之间的摩擦，这种摩擦是在摆动腿通过站立腿时产生的）。
d. 确保在所有断点处，站立脚的接触力都位于摩擦锥内。
e. 确保摆动的脚与地面碰撞产生的冲力位于摩擦锥内。

低动能机器人技术

走路、跑步、游泳、 飞行和操纵的算法

吕斯-特德雷克

最后修改 2021-6-13.

注意：这些是用于在麻省理工学院教授的课程的工作笔记。 它们将在2021年春季学期中被更新。 讲座视频可在YouTube上找到。

上一章 目录 下一章

章节 18

在Colab中打开

系统识别

我在这些笔记中的主要重点是建立算法，以设计和分析一个*给定的*工厂*模型*的控制系统。 事实上，我们在一些地方花了很大的力气去理解我们模型中的结构（特别是操纵器方程的结构），并试图编写利用这种结构的算法。

近年来，我们对机器人的雄心壮志已经发展到了质疑这一假设的程度。 如果我们想给机器人编程，让它叠衣服、在吐司上涂花生酱或做沙拉，那么我们绝对不应该假设我们只是被赋予一个模型（以及估计该模型状态的能力）。 这导致许多研究人员关注强化学习中流行的 "无模型 "最优控制方法。 但我担心，纯粹的无模型方法是 "把婴儿和洗澡水一起扔掉"。 我们有非常棒的工具，可以在给定模型的情况下做出长期决策；而无模型的方法则相应地要弱得多。

所以在这一章中，我想介绍一下学习模型的问题。 这远远不是一个新问题。 系统识别 "领域和控制本身一样古老，但机器学习的新成果极大地增加了我们的算法和分析，特别是在高维和零星样本制度下。 但在最近的机器学习热潮之前，系统识别作为一个领域有一个非常强大的基础，对基本算法有透彻的统计理解，至少在渐进制度（数据量达到无限大的极限）。 我在这一章的目标是建立这个基础，并提供一些指导，但我将停止在这里提供完整的统计学观点。

18.1 问题制定：方程错误与模拟错误

我们的问题表述不可避免地从数据开始。在实践中，如果我们能接触到一个物理系统，并使用数字电子仪器，那么我们就有一个系统，在这个系统中我们可以以某种离散的速率应用输入指令$\mathbf{u}_n$，并以某种离散的速率测量系统的输出$\mathbf{y}_n$。我们通常假定这些速率是固定的，并经常试图建立一个状态空间模型，其形式为

$$\mathbf{x}[n+1] = f_\alpha(\mathbf{x}[n], \mathbf{u}[n])。 \qquad \mathbf{y}[n] = g_\alpha(\mathbf{x}[n], \mathbf{u}[n])。 \tag{1}$$

在这里我再次使用$\boldsymbol{\alpha}$来表示参数向量。在这种情况下，一个自然的表述是最小化最小二乘法的估计目标。

$$\min_{\alpha, \mathbf{x}[0]} \sum_{n=0}^{N-1} \|\mathbf{y}[n] - \mathbf{y}_n\|_2^2, \qquad 受制于(1).$$

我已经写了纯粹的确定性模型来开始，但在一般情况下，我们希望状态的演变和测量都有随机性。有时，就像我写的那样，我们将确定性模型与数据对接，并依靠我们的最小二乘法目标来捕捉误差；更普遍的情况是，我们会研究将随机模型与数据对接。

我们通常把识别程序分成两部分，首先估计给定n输入输出数据$\mathbf{u}_n$、$\mathbf{y}$的n状态$\hat{\mathbf{x}}$，然后在第二步集中估计状态演化的动力学。动态估计的算法主要分为两类。

- *方程误差*只使一步预测误差最小化。

$$\min_{\alpha} \sum_{n=0}^{N-2} \|f_\alpha(\hat{\mathbf{x}}_n, \mathbf{u}_n) - \hat{\mathbf{x}}_{n+1}\|_2^2。$$

- *模拟误差*捕捉到了长期预测误差。

$$\min_{\alpha} \sum_{n=1}^{N-1} \|\mathbf{x}[n] - \hat{\mathbf{x}}_n\|_2^2。 \qquad \tau x[n+1] = f_\alpha(\mathbf{x}[n], \mathbf{u}_n), \mathbf{x}[0] = \hat{\mathbf{x}}_0,$$

方程误差公式通常会导致更多可操作的优化问题，但不幸的是，我们将看到优化一步误差仍然会导致任意大的模拟误差。因此，我们通常认为模拟误差是我们希望优化的真正目标，而方程误差只是一个潜在的有用的替代物。

18.2 机械的参数识别系统

我在这些笔记中主要关注的是（欠驱动的）机械系统，但是当涉及到识别时，有一个重要的区别要做。对于一些机械系统，我们知道模型的结构，包括状态变量的数量和运动学树的拓扑结构。像Spot或Atlas这样的有腿机器人就是很好的例子 -- 动力学肯定是不简单的，但方程的一般形式是已知的。在这种情况下，识别的任务实际上是估计结构化模型中的参数的任务。这就是

本节的主题。

叠衣服或做沙拉的例子则属于另一个类别。在这些例子中，我可能甚至不知道提供一个合理的行为描述所需的状态变量的数量。这将迫使我们对识别问题进行更广泛的研究，我们将在剩下的章节中探讨这个问题。

让我们从确定一个典型的欠驱动机械系统的问题开始，比如说Acrobot、Cart-Pole或Quadrotor，我们知道方程的结构，但只需要Øt参数。我们将进一步假设，我们有能力直接观察所有的状态变量，尽管是有噪声的测量（例如来自关节传感器和/或惯性测量单元）。我们即将讨论的更强大的状态估计算法假设了一个模型，所以我们通常不在这里直接使用它们。

在我们继续之前，请考虑花一分钟时间回顾一下推导双摆的操纵器方程的例子。

18.2.1 运动学参数和校准

我们可以把多体方程中的参数再次分成运动学参数和动力学参数。运动学参数，如链接长度，描述了运动学树中一个关节到另一个关节的坐标转换。当然，写一个优化程序来校准这些参数是可能的；你可以在[]的第11章中找到相当全面的讨论。1].但我想我一般认为，如果你对你的链接长度没有准确的估计，那么你可能应该在投资非线性优化之前投资一个卷尺。

其中一个明显的例外是关于联合oÆsets的校准。这一点在实践中可能是一个真正的麻烦。关节传感器可能会打滑，有些机器人甚至使用相对旋转编码器，并依靠在每次机器人通电时将关节驱动到某个已知的硬关节极限，以获得oÆset。我曾在一个人形机器人上工作过，它有一个相当复杂和痛苦的运动学校准程序，包括在关节上安装额外的硬件，以确保它们在一个已知的位置，然后运行一个脚本。有一个昂贵的和/或不可靠的校准程序会给任何机器人项目带来阻碍。特别是对于欠驱动的系统，它可以对性能产生巨大的影响。

> **例子（有校准误差的18.1Acrobot平衡）。**
>
> 当试图稳定像Acrobot这样的系统时，小的运动学校准误差会导致大的稳态误差。我把一个简单的笔记本放在一起，以显示这里的效应。
>
> 在Colab中打开
>
> 我们来自稳健/随机控制的工具非常适用于识别（和约束/最小化）这些敏感性，至少对于我们在LQR中使用的线性化模型是如此。

从数据中估计关节oÆsets的一般方法是将关节oÆsets作为一个参数来写方程，例如对于双摆，我们将把前向运动学写成：

$$\mathbf{p}_1 = l_1 \begin{bmatrix} \sin(\vartheta_1 + \bar{\vartheta}_1) \\ -\cos(\vartheta_1 + \bar{\vartheta}_1) \end{bmatrix}。 \quad \mathbf{p}_2 = \mathbf{p}_1 + l_2 \begin{bmatrix} \sin(\vartheta_1 + \bar{\vartheta}_1 + \vartheta_2 + \bar{\vartheta}_2) \\ -\cos(\vartheta_1 + \bar{\vartheta}_1 + \vartheta_2 + \bar{\vartheta}_2) \end{bmatrix}。$$

我们试图获得独立的端点位置测量值（例如从运动捕捉，也许从一些机器人安装的摄像机，或从一些机械校准装置），以及它们相应的关节测量值，以获得4$\mathbf{p}_2, \vartheta_1, \vartheta_2$5形式的数据点。然后我们可以解决一个小型的非线性优化问题，以估计关节的oÆsets，使最小的残差最小。

如果没有独立的运动学测量数据，就可以用三角函数的方法，如$s_{\vartheta+\bar{\vartheta}}=_{\vartheta}sc_{\bar{\vartheta}}+cs_{\vartheta\bar{\vartheta}}$，来估计oÆsets和动态参数，然后将$s_{\bar{\vartheta}}$、$c_{\bar{\vartheta}}$项（单独）纳入我们下面讨论的 "包干参数"。

18.2.2 最小平方变换（反动力学）。

现在让我们来考虑估计多体系统的动态参数。我们已经在写操纵方程的形式。

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} = \tau(_{g}\mathbf{q}) + \mathbf{Bu} + \text{摩擦}, \text{ 等等。} \quad (2)$$

这个方程中的每个项都可以取决于我们试图估计的参数α。但参数是以一种特殊的结构方式进入多体方程的：方程是*以包络参数为基础的*。更确切地说，上面的操纵器方程可以被分解为以下形式

$$\mathbf{W}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{u})\alpha_{l}(\alpha) + \mathbf{w}(_{0}\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{u}) = 0,$$

其中α_l是 "包络参数"。我们有时将$\mathbf{W}$称为 "数据矩阵"。

例子18.2(简单钟摆的结点参数)

现在人们熟悉的单摆的运动方程是

$$ml^2\ddot{\vartheta} + b\dot{\vartheta} + mgl\sin\vartheta = \tau$$

[$\ddot{\vartheta}$ $\dot{\vartheta}$ 正弦]。ϑ mgl) $-\tau = 0.$

对于参数估计，我们将把这个因素纳入术语ml^2、b和mgl共同构成了 "包络参数"。

值得花点时间来反思一下这个因式分解。首先，它确实代表了关于多体方程的某种特殊声明：非线性只以特定方式进入。例如，如果我在方程中有sin($m\vartheta$)这样的条款，那么我将无法产生一个将m和ϑ分开的aÆine分解。幸运的是，这种情况在我们的机械系统中并没有发生[...1].此外，这种结构对于*逆向动力学*是特别的，正如我们在这里写的那样。如果你要写正向动力学。
乘以$\mathbf{M}^{1\text{-}}$，以求解$\mathbf{q}$ 那么你将再次破坏这个
aÆine结构。

这真是太有趣了!以我们的标准状态空间形式来考虑一般动力系统的参数估计是很诱人的：$\mathbf{x}[n+1]=f_{\alpha}(\mathbf{x}[n], \mathbf{u}[n])$。但是对于多体系统来说，这似乎是错误的，因为它破坏了这个美丽的阿埃尼结构。

例子 (DRAKE中的18.3多体参数)

很少有机器人模拟器能让你访问动力学的参数。在德雷克，我们在一个单独的数据结构中明确声明多体系统的所有参数，使它们可用，我们可以利用德雷克的符号引擎来提取和操作与这些变量有关的方程。

作为一个简单的例子，我从URDF中加载了车杆系统模型，创建了一个符号版本的MultibodyPlant，并在Context中填充了感兴趣的量的符号变量。然后我可以评估（反）动力学，以获得我的方程。

在Colab中打开

输出结果看起来像。

符号化的动力学。

```
(0.10000000000000001 * v(0) - u(0) + ( pow(v(1),2 )* mp * l * sin(q(1)))+ (vd(0) * mc) +
(0.10000000000000001 * v(1) - (vd(0) * mp * l * cos(q(1) ) )+ (vd(1) * mp * pow(l,2 ))+ 9.
```

继续将这些与我们手工推导的车杆方程式进行比较。

Drake提供了一个方法DecomposeLumpedParameters，该方法将接受这个表达式，并将其分解成上面的aÆine表达式。 对于这个车极的例子，它提取了包络参数[$m_c + m_p, p\ ml, p\ ml^2$]。

肿块参数分解的存在揭示了肿块参数估计的*方程误差*，其误差取自扭矩坐标，可以用最小二乘法解决。 因此，我们可以利用线性估计的所有强大结果和变体。例如，我们可以添加条款来规范化估计（例如，保持接近初始猜测），我们可以编写eÆicient递归估计器，使用递归最小二乘法对参数进行最佳在线估计。我最喜欢的递归最小二乘法算法使用增量QR分解法[2].

重要的是，由于我们已经将其简化为一个最小二乘法问题，我们也可以理解它在什么时候会*失效*。特别是，很可能有些参数无法从机器人上的任何数量的关节数据中估计出来。作为一个简单的例子，考虑一个用螺栓固定在桌子上的机器人手臂；机器人第一环节的惯性参数将无法从任何数量的关节数据中识别。即使在第二个环节上，也只有相对于第一个关节轴的惯性是可识别的；与其他尺寸相对应的惯性参数是不可识别的。在我们的最小二乘法公式中，这很容易理解：我们只需检查数据矩阵的（列）等级，$\mathbf{W}$。特别是，我们可以通过使用，例如，$\mathbf{R}_1\alpha_l$从QR分解中提取**可识别的块状参数**。

$$\mathbf{W}=\begin{bmatrix}\mathbf{Q}_1 & \mathbf{Q}_2\end{bmatrix}\begin{bmatrix}\mathbf{R}_1\\ 0\end{bmatrix}, \quad \Rightarrow \quad \mathbf{W}\alpha_l=\mathbf{Q}_1(\mathbf{R}_1\alpha_l)。$$

例子 (18.4Cart-Pole的参数估计)

在从上面的URDF Øle中提取了集合参数后，我们现在可以将其付诸实施。我让这个例子保持简单。我在输入端模拟了几秒钟的车极，只运行了简单的正弦波轨迹，然后构建了数据矩阵并进行了最小二乘法Øt。

在Colab中打开

输出看起来像这样。

```
估算的整数参数。
(mc + mp)。URDF:11.0，估计。 10.905425349337081 (mp *
l).URDF:0.5，估计。 0.5945797067753813
(mp * pow(l,2 ))。URDF。0.25，估计的。 0.302915745122919
```

请注意，我们可以用更多的数据（或更精心挑选的数据）轻松地使Øt更加准确。

我们是否应该对只估计（可识别的）凑合参数感到满意？我们追求的不是真正的原始参数吗？数据矩阵的线性代数分解（假设我们把它应用于一个丰富的数据集），实际上是为我们揭示了关于我们系统动态的一些基本情况。我们不应该因为无法准确估计某些参数而感到失望，而是应该接受我们不*需要估计这些参数*来进行任何关于我们方程的动态推理（模拟、验证、控制设计等等）。可识别的包络参数正是我们需要的包络参数的子集。

出于实际的原因，可能方便的做法是采取你对包络参数的估计，并尝试回溯原始参数（例如，如果你想把它们写回URDF Øle中）。为此，我建议有一个Ønal后处理步骤，例如Ønds参数$\boldsymbol{\alpha}$尽可能地接近（例如在最小二乘法意义上）你对参数的原始猜测，受制于$\mathbf{R}_1\boldsymbol{\alpha}_l(\boldsymbol{\alpha})$与估计的可识别包络参数匹配的非线性约束。

还有一些微妙之处值得考虑，比如我们如何对惯性矩阵进行参数化。直接估计天真的参数化，即一个对称的3x3矩阵的六个条目，会导致非物理的惯性矩阵。[3] 描述了一个参数估计公式，其中包括这些参数之间的物理性约束的凸公式。

18.2.3 使用能量而不是逆向动力学进行识别。

除了利用线性代数的工具外，还有一些对基本配方的重新调整，以利用我们对机械学的理解。一个重要的例子是参数估计的 "能量公式"[4].

我们已经观察到，在扭矩空间（逆向动力学）中评估方程误差可能比在状态空间空间（正向动力学）中评估要好，因为我们可以将逆向动力学因素化。但这一特性并不专属于逆动力学。系统的总能量（动能+势能）也是包络参数中的一个Æine。我们可以用这样的关系

$$\dot{E}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) = \dot{\mathbf{q}}^T \ (\mathbf{Bu} + \text{friction}, \quad ...).$$

为什么我们喜欢用能量坐标而不是扭矩来工作？两者的区别在详细的数值计算中是显而易见的。在扭矩公式中，我们发现自己直接使用$\ddot{\mathbf{q}}$。传统的观点认为，关节传感器可以可靠地报告关节位置和速度，但关节加速度，通常是通过*两次*数值偏离得到的，往往噪音更大。在某些情况下，对系统的总能量而不是单个关节的能量应用Ønite diÆerences可能在数值上更好。[5]对各种Øltering公式进行了研究，并在[]中提出了建议。4]中建议采用能量公式

† 有时这些方法被写成

倾向于在数量上占优势。

18.2.4 残余物理学模型与线性函数的近似值

最近，"残余物理学 "这个词变得相当流行（例如[.6]），因为人们正在寻找将深度神经网络的建模能力与我们最好的机械学工具结合起来的方法。但实际上，这个想法是相当古老的，有一类自然的残差模型可以很好地适用于我们的最小二乘法凑合参数估计。具体而言，我们可以考虑以下形式的模型。

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} = \tau(_g\mathbf{q}) + \alpha_r\mathbf{\Phi}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{Bu} + \text{friction}, \quad \text{等.}, \tag{3}$$

α_r是残差的附加参数，$\mathbf{\Phi}$是一组（Øxed）非线性基函数。希望这些残差模型能够捕捉到动力学中任何可预测的、但我们没有包括在原始参数模型中的 "滑移项"。非线性摩擦和空气动力阻力是经常被引用的例子。

这些基函数，$\Phi(\mathbf{q}, \dot{\mathbf{q}})$，用于操纵器方程的常见选择包括径向基函数[7] 或小波[8].尽管允许基函数依赖于$\ddot{\mathbf{q}}$或$\mathbf{u}$会对参数估计产生影响，但我们倾向于将其限制在$\mathbf{q}$和$\dot{\mathbf{q}}$，以保持操纵器方程的一些其他良好特性（如控制-aÆine）。

由于最小二乘法估计的成熟性，也有可能使用最小二乘法来有效地确定有效描述系统动力学的基函数子集。例如，在[9]中，我们将最小二乘法应用于各种物理启发的基函数，以便对栖息过程中失速后的空气动力学进行更好的ODE近似，最终除了少数最能描述数据的基函数外，我们放弃了所有基函数。现在，我们可以应用像LASSO这样的算法来进行最小二乘回归，并将其与
ℓ-1regularization，或者[.....]使用基于顺序阈值最小二乘法的替代方法。10]则采用了一种基于顺序阈值最小二乘法的替代方法。

18.2.5 实验设计是一种轨迹优化

对于任何有关我们的参数估计算法恢复真正的可识别包络参数的说法，一个关键的假设是数据集足够丰富；轨迹是 "参数激动 "的。基本上，我们需要假设轨迹产生的运动使数据矩阵$\mathbf{W}$包含了所有可识别的包络参数的信息。由于我们的线性参数化，我们可以通过$\mathbf{W}$上的数字线性代数来评估这一点。

此外，如果我们有机会改变机器人在为参数估计收集数据时执行的轨迹，那么我们可以设计试图最大限度地激发参数的轨迹，并产生一个数值条件良好的最小二乘法问题。一个自然的选择是最小化数据矩阵的条件数，$\mathbf{W}$。
矩阵的最大奇异值与最小奇异值的比率为$\frac{\sigma_{max}(\mathbf{W})}{\sigma_{min}(\mathbf{W})}$。条件是
根据定义，数字总是大于1，数值越小，条件越好。数据矩阵的条件数是整个轨迹上所取数据的非线性函数，但它仍然可以在非线性轨迹优化中被优化（[1], § 12.3.4).

18.2.6 在线估计和自适应控制

适应性控制的领域是一个巨大而丰富的文献；许多书籍都是关于这个主题的（例如[11]).请允许我在此简单介绍一下

这里的文献。

到目前为止，我主要讨论的是参数估计，作为一个在线程序，但自适应控制的一个重要方法是在控制器执行时对参数进行在线估计。由于我们的估计目标可以是线性的（整数）参数，这通常相当于递归最小二乘法估计。为了正确分析这样的系统，我们可以认为该系统具有一个增强的状态空间，$\bar{\mathbf{x}} = [\mathbf{x}$，并研究以下的闭环动态 α

态和参数的共同演化。在机器人操纵器的自适应控制方面，一些最有力的成果是关于全动操纵器的，但例如[12]给出了一个很好的例子，用我们在这些笔记中开发的许多工具分析了欠驱动系统的自适应控制器。

正如我所说，自适应控制是一个丰富的主题。然而，从那个Øeld中得到的最大教训之一是，为了实现任务，可能不需要实现对真实（凑合）参数的收敛。自适应控制中的许多经典结果对任务的执行作出了保证，但明确地不要求/保证参数的收敛。

18.2.7 识别与联系

我们能否将这些同样的技术应用于例如正在与环境进行接触和断开接触的行走机器人？

当然，这个问题有一个版本可以立即发挥作用：如果我们知道接触雅各布，并且有接触力的测量值，那么我们可以将这些项直接添加到操纵器方程中，并继续对包络参数进行最小二乘估计，甚至包括摩擦参数。

我们还可以研究接触力不能直接测量的情况。例如，[13]研究了有无明确接触力测量的被动物体的惯性参数的可识别性的极端情况。

18.3 识别（时域）线性动力系统

如果说多体参数估计构成了系统识别工作的第一个相关支柱，那么线性系统的识别则构成了第二个支柱。自该领域成立以来，线性模型一直是系统识别研究的主要焦点，在过去几年中，由于机器学习的新成果贡献了新的界限和收敛结果，特别是在零星样本制度中，线性模型又重新流行起来（例如[.14, 15, 1617]).

线性系统识别文献的一个重要部分（例如[.18]）集中在识别频域中的线性模型。的确，传递函数的实现提供了重要的洞察力，并避免了我们需要解决的状态空间实现的一些缺陷。然而，在本章中，我将把注意力集中在线性动力系统的时域描述上；这里的一些经验更容易被推广到非线性动力上（另外，不幸的是，在这些笔记中，我们还没有建立起频域技术的基础）。

18.3.1 从国家观察来看

让我们从简单的情况开始处理。从对状态的直接（可能是有噪声的）测量中确定一个线性模型。拟合一个离散时间模型。

$\mathbf{x}[n+1] = \mathbf{A}\mathbf{x}[n] + \mathbf{B}\mathbf{u}[n] + \mathbf{w}[n]$，对采样数据（$\mathbf{u}_n, \mathbf{x}_n = \mathbf{x}[n] + \mathbf{v}[n]$）使用*方程误差*目标，只是另一个*线性*最小二乘问题。通常情况下，我们形成一些数据矩阵，并将我们的最小二乘问题写成。

$$\mathbf{X}' \approx [\mathbf{A}\ \mathbf{B}] \begin{bmatrix} \mathbf{X} \\ \mathbf{U} \end{bmatrix}$$,其中

$$\mathbf{X} = \begin{bmatrix} \\ \\ \end{bmatrix}_{x0\ x1\ \cdots\ xN-2}, \quad \mathbf{X}' = \begin{bmatrix} \\ \\ \end{bmatrix}_{x1\ x2\ \cdots\ xN-1}, \quad \mathbf{U} = \begin{bmatrix} \\ \\ \end{bmatrix}_{u0\ u1\ \cdots\ uN-2}.$$

根据线性最小二乘法的优点，这个估计器对于不相关的过程和/或测量噪声$\mathbf{w}[n]$和$\mathbf{v}[n]$是无偏的。

例18.5（用一个确定的模型进行车极平衡）。

我提供了一个笔记本，展示了线性识别在车杆系统中的实际应用，分为一系列的步骤，看起来像什么。首先，我设计了一个LQR平衡控制器，但对车杆使用了*错误的*参数（我改变了质量等）。这个LQR控制器足以保持车杆不倒，但它不能很好地收敛到直立状态。我想问的是，我能否用这个实验产生的数据来确定一个围绕Øxed-point的更好的线性模型，并改进我的LQR控制器？

有趣的是，简单的答案是 "不"。如果你只收集运行这个控制器的输入和状态数据，你会发现我们上面制定的最小二乘法问题是等级缺失的。估计的$\mathbf{A}$和$\mathbf{B}$矩阵，表示为$\hat{\mathbf{A}}$和$\hat{\mathbf{B}}$ 描述了数据，但没有揭示出真正的线性化。 如果你根据这些估计值设计一个控制器，你将会很失望!

幸运的是，我们可以通过检查最小二乘法解的等级看到这个问题。生成更多的例子并不能Øx这个问题。相反，为了生成更丰富的数据集，我在输入中加入了一个小的额外信号：$u(t) = \pi_{lqr}(\mathbf{x}(t)) + \sin(0.1t)$。这就产生了所有的差异。

在Colab中打开

我希望你能试试这段代码。估算的基本算法简单得令人吃惊，但要使它真正发挥作用，还有很多细节需要把握。

基于模型的迭代学习控制（ILC）

例子（18.6流体动力车杆）

我最喜欢的基于模型的ILC的例子之一是一系列的实验，在这些实验中我们探索了 "流体力学车杆 "系统的动力学。可以把它看作是经典的车杆系统和Øghter喷气机之间的交叉体（也许更接近车杆）！在这个实验中，我们发现了一个 "水动力车杆 "系统。

在这里，我们用一个气翼（水翼）取代了杆子，把整个系统翻转过来，并把它扔进一个水隧道。与其说是摆动起来平衡杆子对抗重力，不如说是平衡处于不稳定状态下的翼片对抗水动力。这些力是不稳定的液-体相互作用的结果；与经典的车-杆系统不同，这次我们没有一个可操作的参数化的系统ODE模型。这是一个用于系统识别和ILC的完美问题。

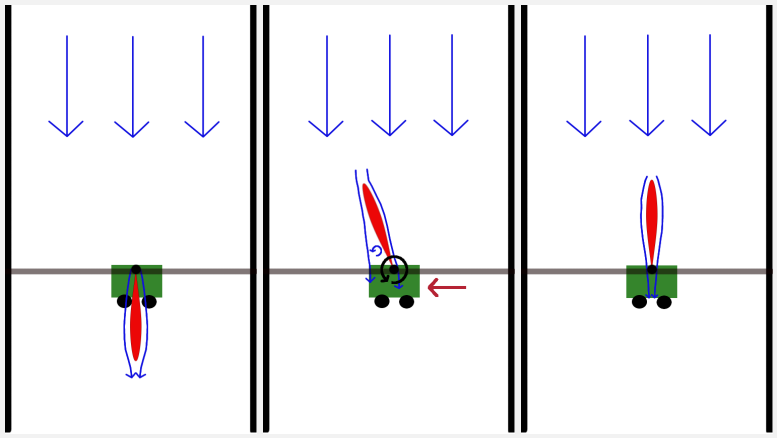


图-18.2-水动力车-杆系统的卡通。小车在水平方向上被驱动，气膜围绕着一个被动关节转动，液体沿箭头方向流动。(整个系统是水平的，所以没有重力的影响。)机翼的空气动力中心在机翼中间的某处；由于针状接头在尾部，被动系统将 "风向标 "到稳定的 "向下 "平衡（左）。平衡对应于稳定不稳定的 "向上 "平衡（右）。过渡期（中间）的液体动力学是不稳定的，非常非线性的。

在一系列的实验中，我们首先尝试使用一个用近似模型（使用∞平板理论）得出的LQR控制器来稳定系统。这个控制器的性能并不理想，但却足以收集不稳定的Øxed点附近的相关数据。然后，我们建立了一个线性模型，使用该模型重新计算LQR控制器，并得到了明显更好的性能。

为了研究更积极的机动性，我们考虑对小车的期望位置进行快速的阶跃变化（就像Øghter jet快速改变高度）。仅仅使用具有移动设定点的时间不变的LQR平衡控制器，我们自然观察到一个非常缓慢的阶跃反应。在平衡中使用的时间不变的线性模型上使用轨迹优化，我们可以做得更好。但我们通过在该轨迹附近迭代Øtting一个时变线性模型并执行基于模型的ILC，取得了相当好的性能。

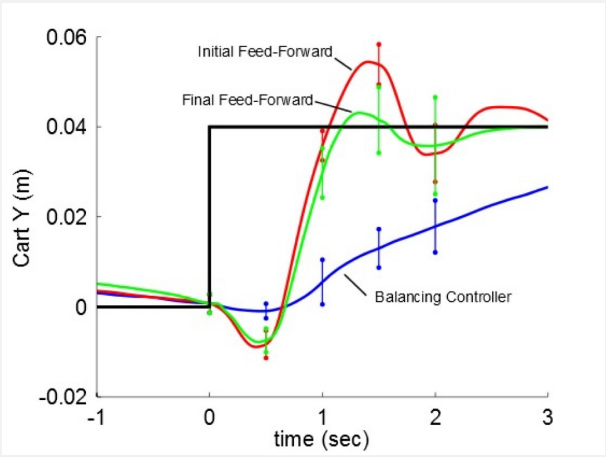


图18.3 - 使用三种不同控制器的阶跃响应比较：平衡的LQR控制器（蓝色），带有LTI模型的轨迹优化得到的前馈项的LQR（红色），以及通过ILC得到的带有时变线性模型和iLQR的控制器（绿色）。

这些实验相当漂亮，非常直观。他们在此基础上继续考虑了用以下方法对进入的涡流干扰进行稳定的效果

对即将到来的液体的实时感知。如果你有任何兴趣，我鼓励你去看看约翰-罗伯特的论文[19] 和/或他的论文答辩视频。

利用主导特征模式进行压缩

对于高维状态或输入向量，我们可以使用奇异值分解（SVD）来解决这个最小二乘问题，只需使用**A**的优势特征值（和相应的特征向量）[...]。20，第7.2节] 使用所谓的 "动态模式分解"（DMD）。已经有许多经验性的成功案例，使用少量的主导模式就能产生一个很好的数据Øt（这在像液体动力学这样的问题中尤其重要，因为状态向量**x**可能是对应于液体∞流的整个图像）。在DMD中，我们将在这些特征模的坐标中写出线性动力学（它总是可以被投射回全坐标）。

在非线性基础上的线性动力学

我们在这里可以考虑的一个潜在的有用的概括是确定在由一些（可能是高维的）基向量$j(\mathbf{x})$定义的坐标系中线性发展的动力学。特别是，我们可以考虑$\dot{j} = \mathbf{x}^{\phi\cdot} = Aj(x)$的动力学形式。 由于与Koopman算子理论的联系，人们对这种特殊的形式大加赞赏，我们将在下文详细讨论。为了我们的目的，我们还需要一个控制输入，所以可以考虑$\dot{j} = Aj(x) + \mathbf{Bu}$这样的形式。

再次，由于最小二乘法的成熟，用这种方法可以列出许多可能的基函数，然后用稀疏最小二乘法和/或模态分解来有效地找到重要的子集。

请注意，上面描述的多体参数估计不是这样的，尽管它是密切相关的。操纵器方程的最小二乘法总括参数估计发现了在状态变量中仍然是*非线性的*动力学。

18.3.2 从输入-输出数据（状态实现问题）来看

在更一般的形式下，我们希望估计一个形式为

$$\mathbf{x}[n+1] = \mathbf{A}\mathbf{x}[n] + \mathbf{B}\mathbf{u}[n] + \mathbf{w}[n] 。$$
$$\mathbf{y}[n] = \mathbf{C}\mathbf{x}[n] + \mathbf{D}\mathbf{u}[n] + \mathbf{v}[n]。$$

我们将再次应用最小二乘法估计，但将其与著名的 "Ho-Kalman "算法（也称为 "Eigen System Realization"（ERA）算法）相结合[21].我最喜欢的关于这个算法的介绍是[16].

首先，观察一下

$$\mathbf{y}[0] = \mathbf{C}\mathbf{x}[0] + \mathbf{D}\mathbf{u}[0] + \mathbf{v}[0] 。$$
$$\mathbf{y}[1] = \mathbf{C}(\mathbf{A}\mathbf{x}[0] + \mathbf{B}\mathbf{u}[0] + \mathbf{w}[0]) + \mathbf{D}\mathbf{u}[1] + \mathbf{v}[1] 。$$
$$\mathbf{y}[n] = {}^{n}\mathbf{C}\mathbf{A}\mathbf{x}[0] + \mathbf{D}\mathbf{u}[n] + \mathbf{v}[n] + \sum_{k=0}^{n-1}\mathbf{C}\mathbf{A}({}^{n-k-1}\mathbf{B}\mathbf{u}[k] + \mathbf{w}[k])。$$

为了识别的目的，让我们把$y[n]$写成一个最近的函数。
N+1输入（对于$k \geq N$）。

$$\mathbf{y}[n] = [CAN1\text{-}\mathbf{B}\ CAN2\text{-}\mathbf{B}\ \nearrow\ \mathbf{CB}\ \mathbf{D}] \begin{pmatrix} \mathbf{u}[n-N] \\ \mathbf{u}[n-N+1] \\ \vdots \\ \mathbf{u}[n] \end{pmatrix} + \delta[n]$$

$$= \mathbf{G}[n]\bar{\mathbf{u}}[n] + \delta[n]$$

其中$\delta[n]$捕获了初始条件、噪声和（滑动）窗口前的控制输入的剩余项。 $\mathbf{G}[n]=[CAN1\text{-}\mathbf{B}\ CAN2\text{-}\mathbf{B}\ \nearrow\ \mathbf{CB}\ \mathbf{D}]$，$\bar{\mathbf{u}}[n]$代表从时间$n-N$到$n$的串联$\mathbf{u}[n]$'s。 重要的是我们有$\delta[n]$与$\bar{\mathbf{u}}[n]$不相关：$\forall k > n$我们有$E[\mathbf{u}_k \delta_n] = 0$。 这有时被称为Neyman正交性，它意味着我们可以用简单的最小二乘法估计$\hat{\mathbf{G}} = \mathbf{argmin}_{\mathbf{G}} \sum_{n \geq N} |\mathbf{y}_n - \mathbf{G}\bar{\mathbf{u}}_n|^2$。 [16]给出了估计误差的常数与样本数和方差的关系的界限。
噪声。

我们应该如何选择窗口大小N？根据Neyman正交性，我们知道对于任何$N \geq 0$的选择，我们的估计都是无偏的。但是如果我们选择的N太小，那么$\delta[n]$项就会很大，导致我们的估计有可能出现很大的方差。对于稳定的系统，δ项会随着我们增加N而变小。在实践中，我们根据数据中的特征沉淀时间来选择N（大致到脉冲响应变得足够小）。

如果你学习过线性系统，$\mathbf{G}$看起来会很熟悉；它正是这种（多输入、多输出）矩阵脉冲响应，也被称为 "马尔科夫参数"。
事实上，估计$\hat{\mathbf{G}}$ 甚至可能适合控制设计，因为它是密切------的。
与部分可观测系统的基于干扰的反馈中使用的参数化有关[.2223].但是Ho-Kalman算法可以用来提取
从$\hat{\mathbf{G}}$中得到状态维度为$\dim(\mathbf{x})=n$的$_x$良好估计$\hat{\mathbf{A}}$, $\hat{\mathbf{B}}$, $\hat{\mathbf{C}}$, $\hat{\mathbf{D}}$, 假定
真正的系统是可观察和可控制的，至少有n_x个阶，而且我们在下面形成的数据矩阵是足够丰富的[16].

重要的是要认识到，许多系统矩阵，$\mathbf{A}$、$\mathbf{B}$、$\mathbf{C}$，可以描述相同的输入-输出数据。特别是，对于任何可逆矩阵（又称相似性变换）。
$\mathbf{T}$，系统矩阵$\mathbf{A}$、$\mathbf{B}$、$\mathbf{C}$和$\mathbf{A}'$、$\mathbf{B}'$、$\mathbf{C}'$与。

$$\mathbf{a}' = {}^{-1}\mathbf{tat}, \qquad \mathbf{b}' = \mathbf{tb}^{-1}, \qquad \mathbf{c}' = \mathbf{ct}。$$

描述相同的输入-输出行为。Ho-Kalman算法返回一个*平衡的实现*。平衡的实现是指我们有效地应用了相似性变换，即$\mathbf{T}$，它使可控性和可观测性的格拉米数相等且对角[...20第9章]，并按对输入/输出行为的影响递减来排列各状态。这种排序对于确定系统的顺序和模型的减少是有意义的。

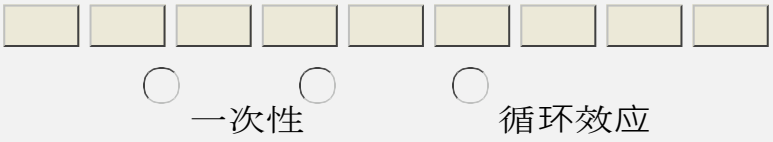
请注意，$\hat{\mathbf{D}}$是$\hat{\mathbf{G}}$中的最后一个区块，所以被提取出来是很简单的。Ho-Kalman算法告诉我们如何提取$\hat{\mathbf{A}}$、$\hat{\mathbf{B}}$、$\hat{\mathbf{C}}$，并在适当构造的数据矩阵上再应用SVD（见例如[16, §5.1], [24, §10.5], 或 [, §9.3]）。20, §9.3]).

例18.7（从关键点识别车杆的Ho-Kalman方法）。

让我们重复一下车杆的例子。但是这一次，我们将考虑从渲染场景的摄像机中识别动态的问题，而不是直接获得对关节位置和速度的观察，但是让我们深思熟虑地进行。

标准相机的输出是一个RGB图像，由3×宽×高的实值组成。我们当然可以将这些列成一个向量，并将其作为输出/观测值$y(t)$。但我并不推荐这样做。这个 "像素空间 "不是一个

好的空间。例如，你可以很容易地找到一个在某一帧中具有小车颜色的像素，但在小车的位置发生递增变化后，现在（不连续地）采用了背景的颜色。深度学习为我们提供了奇妙的新工具，可以将原始的RGB图像转化为更好的 "特征空间"，这将是足够强大的，可以部署在真实的系统上，但可以使我们的建模工作更容易进行。我的小组已经大量使用了 "关键点网络"
[25]和自我监督的 "密集对应"[262728]，将RGB输出转换为更可消费的形式。

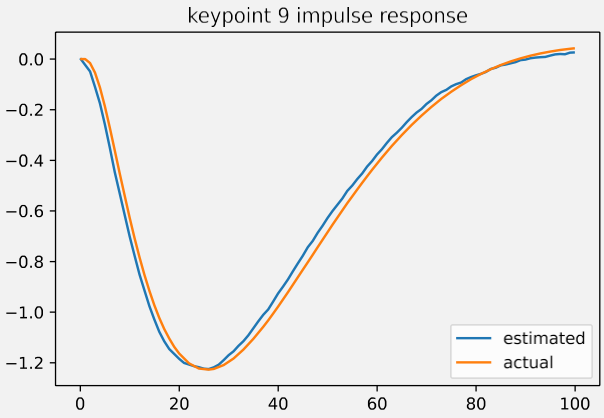


这些工具的存在使得我们有理由假设，我们有代表一些 "关键点 "的二维位置的观测值，这些 "关键点 "被刚性地固定在小车和杆上。例如，任何关键点K在杆子上的位置是由车-杆运动学给出的。

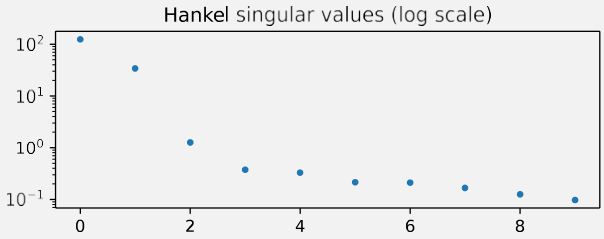
$$ {}^{W}p^{K} = \begin{bmatrix} x \\ 0 \end{bmatrix} + \begin{bmatrix} \cos\vartheta & -\sin\vartheta \\ \sin\vartheta & \cos\vartheta \end{bmatrix} {}^{P}p^{K} 。 $$

其中x是小车的位置，ϑ是杆的角度。 我使用了Monogram符号来表示${}^{P}K_{P}$是点A在杆的框架P中的笛卡尔位置，${}^{K}W$是该点在摄像机/世界框架W中的相同位置。虽然将RGB输出线性化为车杆位置的函数x和ϑ可能是不合理的，但通过线性化这个方程来近似小角度的关键点位置*是*合理的。

在这个例子中，你会看到我们很好地恢复了脉冲响应。



最大的问题是：Ho-Kalman是否发现了我们所知道的、喜欢的四维状态矢量的车杆？它是否发现了一些不一样的东西？我们通常会通过观察汉克尔数据矩阵的奇异值的大小来选择顺序。在这个例子中，在没有噪声的情况下，你可以清楚地看到2个状态描述了大部分的行为，而在45个状态之后，我们的回报就会递减。



作为一个练习，尝试在产生数据的系统中加入过程和/或测量噪声，看看它们如何影响这个结果。

18.3.3 增加稳定性约束

详情即将公布。见，例如[29].

18.3.4 自回归模型

另一类重要的线性模型直接从最近的输入和输出的历史中预测输出。

$$\begin{aligned}\mathbf{y}[n+1] = {}&{}_0\mathbf{A}\mathbf{y}[n] + {}_1\mathbf{A}\mathbf{y}[n-1] + \ldots + {}_k\mathbf{A}\mathbf{y}[n-k] \\ &+ {}_0\mathbf{B}\mathbf{u}[n] + {}_1\mathbf{B}\mathbf{u}[n-1] + \ldots + {}_k\mathbf{B}\mathbf{u}[n-k]\end{aligned}$$

这就是所谓的"自回归模型与外生输入（ARX）"模型。ARX模型的系数可以使用线性最小二乘法从投入产出数据中直接确定。

当然可以把这些模型看成是状态空间模型，我们把Ønite历史$\mathbf{y}$和$\mathbf{u}$（除了$\mathbf{u}[n]$）收集到我们的状态向量中。但这不一定是一个非常有效的表示方法；Ho-Kalman算法可能能够找到一个更小的状态表示。这对于部分可观察的系统尤其重要，因为这些系统可能需要一个很长的历史，k。

例子（18.8一个车极关键点动态的ARX模型）

18.4 识别有限的(Po)Mdps

还有一种情况，我们可以很好地理解：当状态、行动和观察（以及时间）都是离散的时候。记得在我们讨论最优控制和动态编程的最开始，我们使用图搜索和状态空间的离散化来给出一个特别好的价值迭代算法的版本。一般来说，我们可以用条件概率来写*随机动态*。

$$\Pr(s[n+1]|s[n],a[n\quad\quad]),\Pr(o[n]|s[n],a[n])。$$

其中我（再次）用s表示离散的状态，a表示离散的行动，o表示离散的观察。在一切都离散的情况下，这些条件概率可以自然地用矩阵/张量来表示。　　我们将使用张量$T_{ijk}\equiv\Pr(s[n+1]=s_i|s[n]=s_j,a[n]=a_k)$来表示过渡矩阵，$O_{ijk}\equiv\Pr(o[n]=o_i|s[n]=s_j,a[n]=a_k)$表示观察矩阵。

18.4.1 从国家观察来看

与我们对线性动力系统的讨论类似，要理解的第一种情况是我们可以直接获得状态和行动轨迹：$S[\cdot]$，$A[\cdot]$。　这相当于识别一个马尔科夫链或马尔科夫决策过程(MDP)。过渡矩阵可以直接从过渡的统计数据中提取出来。

$$T_{ijk}=\frac{i\text{从}s\text{到}s\text{的转换次数}_j,\ a_k}{\text{来自}s,a\text{的转换总数}^{\circ}}$$

$j\ k$

毫不奇怪，为了使这一估计渐进地收敛于真实的过渡概率，我们需要识别（探索）动态是无规律的（对于每一个状态/行动对都被无限次访问）。

对于大型MDP，类似于我们为线性动力系统描述的模态分解，我们也可以考虑过渡矩阵的因子表示。[即将推出...]

18.4.2 识别隐马尔科夫模型（HMMs）。

18.5 神经网络模型

如果说多体参数估计、线性系统识别和有限状态系统是系统识别的第一大支柱，那么（至少现在）深度学习也许是最后的主要支柱。我把这一节放在线性动力系统一节之后并不是巧合。神经网络是强大的工具，但它们可能有点难以仔细思考。值得庆幸的是，我们将看到线性动力系统的许多基本经验和见解仍然适用于此。特别是，我们可以遵循同样的进展：直接从状态观测中学习动力学函数，从输入输出数据中学习状态空间模型（用递归网络），用前馈网络利用输出历史学习输入输出（自回归）模型。

深度学习工具使我们能够相当可靠地调整神经网络以适应我们的训练数据。主要的挑战在于数据的有效性/泛化，以及在实际设计规划器/控制器的基础上。

在这些具有如此复杂描述的模型上。

18.5.1 生成训练数据

当Øtting丰富的非线性模型（如神经网络）时，一个极其有趣的问题是如何产生训练数据的问题。 你可能有这样的感觉：我们希望数据能对状态空间有一定程度的 "覆盖"；这对欠驱动系统来说尤其具有挑战性，因为我们有动态约束，限制我们在状态空间的轨迹。

对于多体参数估计，我们讨论了使用数据矩阵的条件数作为实验设计的一个明确目标。这是使用*线性*最小二乘法优化回归的一种奢侈，它利用了我们从操纵器方程中获得的关于模型结构的专门知识。不幸的是，在更普遍的非线性最小二乘法与一般函数近似的情况下，我们没有一个类似的概念。这种方法也适用于识别线性动态系统。这里的挑战也许被认为是不太严重的，因为对于无噪声情况下的线性模型，即使是由脉冲响应产生的数据也是足够用来识别的。

*在基于模型的强化学习的*背景下，这个话题最近受到了相当大的关注（例如[30]).广义上讲，在理想情况下，我们希望用于系统识别的训练数据与我们在执行最优策略时遇到的数据分布相匹配。但是，在我们设计好控制器之前，我们无法知道这个分布，（在我们目前的讨论中）这需要我们有一个模型。这是一个典型的 "鸡和蛋 "问题。在大多数情况下，它说明了交错进行系统识别和控制设计的重要性，而不是简单的进行一次识别，然后永远使用模型的概念。

18.5.2 从国家观察来看

18.5.3 来自输入-输出数据的状态-空间模型（递归网络）

18.5.4 输入-输出（自回归）模型

18.6 识别 线性系统 替代方案

18.7 混合系统的识别

18.8 与任务相关的模型

18.9 练习题

练习（18.1具有不同目标函数的线性系统识别）。

考虑一个离散时间线性系统，其形式为

$$x[n+1] = Ax[n] + Bu[n]$$ 。

其中$x[n] \in R^p$和$u[n] \in R^q$是步骤n的状态和控制。系统矩阵$A \in R^{p \times p}$和$B \in R^{p \times q}$是模型的未知参数，你的任务是确定给定的模拟轨迹的参数。使用轨迹
在这个python笔记本中提供的模拟，实现以下问题的解决方案。

a. 通过求解确定模型参数

$$\min_{A,B} \sum_{n=0}^{N-1} \| x[n+1] - Ax[n] - Bu[n] \|_2^2 。$$

b. 通过求解确定模型参数

$$\min_{A,B} \sum_{n=0}^{N-1} \| x[n+1] - Ax[n] - Bu[n] \|_\infty 。$$

c. 通过求解确定模型参数

$$\min_{A,B} \sum_{n=0}^{N-1} \| x[n+1] - Ax[n] - Bu[n] \|_1 。$$

练习（18.2栖息滑翔机的系统识别）。

在这个练习中，我们将使用受物理学启发的基函数来确定一个栖息的滑翔机的非线性动力学。在这个python笔记本中，你需要实现最小二乘法的Øtting，并找出描述滑翔机动力学的最佳基函数集。花点时间浏览一下笔记本，了解其中的代码，然后回答以下问题。书面问题也会在笔记本中列出，以方便你的学习。

a. 通过笔记本中的编码部分进行工作。
b. 我们测试的所有基础构架最多使用3个基础函数来计算单个加速度。如果我们将用于计算单个加速度的基函数数量增加到4个，最小二乘残差就会下降。如果使用更多的基函数，我们可以产生更好的Øt，那么我们为什么要限制自己使用3个基函数？

参考文献

1. W Khalil和E Dombre，"机器人的建模、识别和控制"，Elsevier。2004.

2. M.Kaess and A. Ranganathan and F. Dellaert, "i{SAM}:递增平滑和映射", *IEEE机器人反应*，第24,1365-1378页6,。2008.

3. Patrick M Wensing and Sangbae Kim and Jean-Jacques E Slotine, "线性矩阵不等式用于物理上一致的惯性参数识别。A statistical perspective on the mass distribution", *IEEE Robotics and Automation Letters*, vol. no3,. 60-671,, 2017.

4. Maxime Gautier，"用动力模型动态识别机器人"，*《{IEEE}国际机器人与自动化会议论文集》，第1922--1927卷。国际机器人和自动化会议*，第1922--1927卷，第3页3,。1997.

5. Maxime Gautier, "A comparison of Øltered models for dynamic identifiØcation of robots", *Proceedings of the 35th {IEEE} Conference on Decision and Control.决策与控制会议*，第875-880页1,。 1996.

6. Andy Zeng和Shuran Song和Johnny Lee和Alberto Rodriguez和Thomas Funkhouser，"Tossingbot。学习用残余物理学抛出任意物体"，*IEEE Transactions on Robotics*，第1307-1319页4,36,。 2020.

7. R.M. Sanner和J. E. Slotine, "Gaussian Networks for Direct Adaptive Control", *American 1991Control Conference*, pp.2153-2159, 1991.

8. Robert M Sanner和Jean-Jacques E Slotine，"用于机器人系统自适应控制的结构性动态小波网络"，*国际控制杂志*，第70,405-421页3,。 1998.

9. Warren Hoburg和Russ Tedrake，"{无人机}失速后空气动力学的系统识别栖息"，*AIAA Infotech@Aerospace会议论文集*，4月，[2009.链接]。

10. Steven L Brunton and Joshua L Proctor and J Nathan Kutz, "Discovering governing equations from data by sparse identifiØcation of nonlinear dynamical systems", *Proceedings of the national academy of sciences*, vol. no113,. pp15,.3932-3937。 2016.

11. Karl J. Åström和Björn Wittenmark，"自适应{控制}。 适应性{控制}：{第二次}{版本}",信使公司,四月。 2013.

12. Joseph Moore和Russ Tedrake, "Adaptive Control Design for Underactuated Systems Using Sums-of-Squares Optimization", *Proceedings of American 2014Control Conference (ACC)*, June, [2014.链接]。

13. Nima Fazeli and Roman Kolbert and Russ Tedrake and Alberto Rodriguez, "Parameter and contact force estimation of planar rigid-body undergoing frictional contact", *International Journal of Robotics Research*, vol. 36, no. 13-14, pp.1437-1454, [2017.link]。

14. Moritz Hardt and Tengyu Ma and Benjamin Recht, "Gradient descent learns linear dynamical systems", *arXiv preprint arXiv: 1609.05191,* 2016.

15. Elad Hazan and Holden Lee and Karan Singh and Cyril Zhang and Yi Zhang, "Spectral Øltering for general linear dynamical systems", *arXiv preprint arXiv: 1802.03981,* 2018.

16. Samet Oymak和Necmiye Ozay, "Non-asymptotic identifiØcation of lti systems from a single trajectory", *American 2019control conference (ACC)*, pp.5655-5661, 2019.

17. Max Simchowitz and Ross Boczar and Benjamin Recht, "Learning linear dynamical systems with semi-parametric least squares", *Conference on Learning Theory*, pp.2714-2802, 2019.

18. L.Ljung, "System Identiocation:用户的理论"，Prentice Hall。 1999.

19. 约翰-W-罗伯茨，"用实时粒子图像测速法控制欠驱动的流体系统"，博士论文，麻省理工学院，6月，[2012.链接]。

20. Steven L Brunton和J Nathan Kutz，"数据驱动的科学和工程。机器学习、动态系统和控制"，剑桥大学出版社。 2019.

21. BL Ho和Rudolf E K{'a}lm{'a}n, "EÆective construction of linear state variable models from input/output functions", *at-Automatisierungstechnik*,

vol. no14,.1-12, pp.545-548, 1966.

22. Sadra Sadraddini和Russ Tedrake, "有保证的约束满足的鲁棒输出反馈控制", *在第23届ACM国际混合系统会议论文集。计算和控制*, 第4页12,, [2020.链接]。

23. Max Simchowitz and Karan Singh and Elad Hazan, "Improper learning for non-stochastic control", *Conference on Learning Theory*, pp.3320-3436, 2020.

24. Jer-Nan Juang and Minh Q. Phan, "IdentiØcation and {Control} of {机械}{系统}", 剑桥大学出版社, Aug, 2001.

25. Lucas Manuelli* and Wei Gao* and Peter Florence and Russ Tedrake, "{kPAM:KeyPoint AÆordances for Category-Level Robotic Manipulation}", *arXiv e-prints*, pp.arXiv:1903.06684, Mar, [2019.链接]。

26. Peter R. Florence*和Lucas Manuelli*和Russ Tedrake, "密集物体网：通过和为机器人操纵学习密集视觉物体描述符", *机器人学习会议（CoRL）*, 10月, [2018.链接]。

27. Peter Florence 和 Lucas Manuelli 以及 Russ Tedrake, "Self-Supervised Correspondence in Visuomotor Policy Learning", *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 492-499, April, [2020.链接]。

28. Lucas Manuelli and Yunzhu Li and Pete Florence and Russ Tedrake, "Keypoints into the Future:基于模型的强化学习中的自我监督对应关系", *机器人学习会议（CoRL）*, [2020.链接]。

29. Jack Umenberger 和 Johan W{aa}gberg 和 Ian R Manchester 和 Thomas B Sch{\"o}n, "稳定的线性动力系统的最大似然识别", *Automatica*, 第280-292页96,。 2018.

30. Alekh Agarwal and Nan Jiang and Sham M. Kakade and Wen Sun, "Reinforcement Learning:理论与算法", 在线草稿, 2020年。

低动能机器人技术

走路、跑步、游泳、 飞行和操纵的算法

吕斯-特德雷克

最后修改 2021-6-13.
如何引用这些笔记，使用注释，并给予反馈。

注意：这些是用于在麻省理工学院教授的课程的工作笔记。 它们将在2021年春季学期中被更新。 讲座视频可在YouTube上找到。

章节 19

状态估计

19.1 观察者和卡尔曼滤波器

19.2 递归贝叶斯过滤器

19.3磨光

低动能机器人技术

走路、跑步、游泳、 飞行和操纵的算法

吕斯-特德雷克

最后修改 2021-6-13.
如何引用这些笔记，使用注释，并给予反馈。

注意：这些是用于在麻省理工学院教授的课程的工作笔记。它们将在2021年春季学期中被更新。讲座视频可在YouTube上找到。

上一章 目录 下一章

章节 20
无模式的政策搜索

强化学习（RL）是解决相同的最优控制问题的算法集合，我们通过文本重点关注，但RL文献中真正的瑰宝是随机最优控制问题的（几乎）*黑箱优化算法*。一个算法只有一个与优化问题的 "黑箱 "接口的想法意味着它可以通过试错获得（可能是有噪声的）最优成本的样本，但不能访问基础模型，也不能直接访问完整的梯度信息。

这是一个困难的问题!一般来说，我们不能指望RL算法能像更多的结构化优化那样快速优化，而且我们通常最多只能保证收敛到局部最优。但这个框架是非常通用的，所以它可以应用于我们迄今为止所研究的任何其他算法都无法解决的问题。我最喜欢的强化学习的例子是复杂液体动力学的控制（例如[1]).这些系统通常非常难以建模，或者模型是如此高维和复杂，以至于控制设计无法进行。在这些问题中，通过物理实验中的试错来优化可能会更快。

在本章中，我们将研究一种特殊的强化学习方式，它试图通过明确地对一系列策略进行参数化（例如通过参数向量$\boldsymbol{\alpha}$），然后直接搜索优化长期成本的参数，从而找到好的控制器。对于一个具有我们最喜欢的加法目标形式的随机最优控制问题，这可能看起来像。

$$\min_{\boldsymbol{\alpha}} E\left[\sum_{n=0}^{N} \ell(\mathbf{x}[n], \mathbf{u}[n])\right] \tag{1}$$

其中随机变量是从概率密度中抽取的

$$\mathbf{x}[0] \sim p_0(\mathbf{x})。$$
$$\mathbf{x}[n] \sim p(\mathbf{x}[n] | \mathbf{x}[n-1], \mathbf{u}[n-1])。$$
$$\mathbf{u}[n] \sim p_\alpha(\mathbf{u}[n] | \mathbf{x}[n])。$$

最后一个方程是控制策略的概率表示 -- 在每个时间步的行动$\mathbf{u}$是从一个分布中提取的，这个分布以当前状态$\mathbf{x}$为条件。

当然，控制学界也研究过这样的想法，例如在寻求*极值控制*和*迭代学习控制*的范畴内。我将尽可能地建立联系。

20.1 政策梯度法

RL中策略搜索的标准方法之一是通过评估一些样本轨迹来估计相对于策略参数的预期长期成本的梯度，然后进行（随机的）梯度下降。许多这些所谓的 "策略梯度 "算法都是利用一种叫做*似然比法*的推导，这种方法也许最早在[2]中描述的，然后在REINFORCE算法中得到推广[3].它是基于一个看起来像用对数来估计梯度的技巧；我觉得这个技巧经常被呈现出一种神秘的气息。让我们试着确保我们理解它。

20.1.1 似然比法（又称REINFORCE）。

让我们从一个更简单的对随机函数的优化开始。

$$\min_{\alpha} E\left[\ell(\mathbf{x})\right] \text{ with } \mathbf{x} \sim p_{\alpha}(\mathbf{x})$$

我希望这个符号是清楚的？$\mathbf{x}$是一个随机向量，从分布$p_{\alpha}(\mathbf{x})$中抽取，下标表示分布取决于参数向量α。这个函数的梯度是什么？REINFORCE的推导是

$$\begin{aligned}
\frac{\partial}{\partial \alpha} E\left[g(\mathbf{x})\right] &= \frac{\partial}{\partial \alpha} \int \mathbf{dx}\, g(\mathbf{x}) p_{\alpha}(\mathbf{x}) \\
&= \int \mathbf{dx}\, g(\mathbf{x}) \frac{\partial}{\partial \alpha} p_{\alpha}(\mathbf{x}) \\
&= \int \mathbf{dx}\, g(\mathbf{x}) p_{\alpha}(\mathbf{x}) \frac{\partial \log p_{\alpha}(\mathbf{x})}{\partial \alpha} \\
&= E\left[g(\mathbf{x}) \frac{\partial}{\partial \alpha} \log p_{\alpha}(\mathbf{x})\right].
\end{aligned}$$

为了实现这一点，我们使用了对数的导数。

$$\begin{aligned}
y &= \log u \\
\frac{\partial y}{\partial} &= \frac{\partial 1}{u} \frac{u}{\partial x}
\end{aligned}$$

来写

$$\frac{\partial}{\partial \alpha} p(_{\alpha}\mathbf{x}) = p_{\alpha}(\mathbf{x}) \frac{\partial}{\partial \alpha} \log p_{\alpha}(\mathbf{x})。$$

这表明有一种简单的蒙特卡洛算法来估计政策梯度：抽取N个随机样本$\mathbf{x}_i$，然后估计梯度为

$$\frac{\partial}{\partial \alpha} E\left[g(\mathbf{x})\right] \approx \frac{1}{N} \sum_{i} g(\mathbf{x}_i) \frac{\partial}{\partial \alpha} \log p_{\alpha}(\mathbf{x}_i)。$$

这个技巧在最优控制的情况下甚至更有效。对于Ønite-horizon问题，我们有

$$\begin{aligned}
\frac{\partial^{EN}}{\partial \alpha} \left[\sum_{n=0} \ell(\mathbf{x}[n], \mathbf{u}[n])\right] &= \int \mathbf{dx}[\cdot] \mathbf{du}[\cdot] \left[\sum_{n=0}^{N} \ell(\mathbf{x}[n], \mathbf{u}[n])\right] \frac{\partial}{\partial} p_{\mathfrak{C}}(\mathbf{x}[\cdot], \mathbf{u}[\cdot]) \\
&= E\left[\left(\sum_{n=0} \ell(\mathbf{x}[n], \mathbf{u}[n])\right) \frac{\partial}{\partial \alpha} \log p_{\alpha}(\mathbf{x}[\cdot], \mathbf{u}[\cdot])\right]
\end{aligned}$$

其中$\mathbf{x}[\cdot]$是整个轨迹$\mathbf{x}[0], \ldots, \mathbf{x}[n]$，以及

$$p_\alpha(\mathbf{x}[\cdot], \mathbf{u}[\cdot]) = p_0(\mathbf{x}[0]) \left(\prod_{n=1}^{N} p(\mathbf{x}[n]|\mathbf{x}[n-1], \mathbf{u}[n-1]) \right) \left(\prod_{n=0} p_\alpha(\mathbf{u}[n]|\mathbf{x}[n]) \right) 。$$

取对数，我们有

$$\log p_\alpha(\mathbf{x}[\cdot], \mathbf{u}[\cdot]) = \log p_0(\mathbf{x}[0]) + \sum_{n=1} \log p(\mathbf{x}[n]|\mathbf{x}[n-1], \mathbf{u}[n-1]) + \sum_{n=0}^{Nn} \log p_\alpha(\mathbf{u}[n]|\mathbf{x}[n]) 。$$

只有最后几项取决于α，这就得到了

$$\frac{\partial}{\partial \alpha} E\left[\sum_{n=0} \ell(\mathbf{x}[n], \mathbf{u}[n])\right] = E\left[\left(\sum_{n=0} \ell(\mathbf{x}[n], \mathbf{u}[n])\right)\left(\sum_{n=0} \frac{\partial}{\partial \alpha} \log p_\alpha(\mathbf{u}[n]|\mathbf{x}[n])\right)\right]$$

$$= E\left[\sum_{n=0} \left(\ell(\mathbf{x}[n], \mathbf{u}[n]) \sum_{k=0}^{n} \frac{\partial}{\partial \alpha} \log p_\alpha(\mathbf{u}[n]|\mathbf{x}[n])\right)\right]。$$

其中最后一步使用的事实是：$E\left[\ell(\mathbf{x}[n], \mathbf{u}[n]) \log p_\alpha(\mathbf{u}[k]|\mathbf{x}[k])\right] = 0$ for all $k > n$
.

这个更新应该让你吃惊。它说我可以通过只取政策的梯度来确定长期成本的梯度……但不是工厂的梯度，也不是成本的梯度！直觉是，我们可以通过评估政策沿着闭环系统的一些（随机）轨迹滚动来获得梯度，评估每一个成本，然后增加政策中采取与较低长期成本相关的行动的概率。

这个推导经常被作为政策梯度推导提出。这个特征当然是正确的，但我更希望你把它看作是获得政策梯度的一种方法。这是一个特别聪明的方法，因为它使用的信息正是在强化学习中，我们可以获得瞬时成本，$\ell(\mathbf{x}[n], \mathbf{u}[n])$，以及策略，所以允许采取策略的梯度，但不需要对植物模型有任何了解。虽然它很聪明，但并不特别有效 -- 预期值的蒙特卡洛近似值有很高的方差，所以需要很多样本才能得到准确的估计。 其他推导方法也是可能的，有些甚至
更简单，而其他的则是利用植物梯度，如果你能获得这些梯度的话，这些梯度在同位素样本近似中会有不同的表现。

在本节的剩余部分，我想深入了解一下，并试图更好地理解随机更新的性质，以帮助你理解我的意思。

20.1.2 样品有效性

让我们退一步，更广泛地思考如何在黑箱（无约束）优化中使用梯度下降。假设你有一个简单的优化问题。

$$\min_\alpha g(\alpha)。$$

而你可以直接评估$g()$，但不能$\frac{\partial g}{\partial \alpha}$。你如何进行梯度血统？

估计梯度的一种标准技术，代替了分析性梯度信息，是无差异法[.4].估计梯度的Ønite diÆerences方法包括对每个维度的输入参数进行相同的小扰动ε，并使用。

$$\frac{\partial g}{\partial \alpha_i} \approx \frac{g(\alpha + \varepsilon_i) - g(\alpha)}{\varepsilon}。$$

其中ε_i是列向量，ε在第i行，其他地方为零。有限二乘法在计算上非常昂贵，每一个梯度步骤都需要对函数进行$n+1$评估，其中n是输入矢量的长度。

如果每次函数求值都很昂贵呢？也许这意味着拿起一个物理机器人，为$g()$的每一次评估运行10秒钟。突然间，试图用最少的$g()$评估次数来优化成本函数就变得很重要了。这就是强化学习中的游戏规则--它通常被称为RL的*样本复杂性*。我们能不能用更少的求值进行梯度下降？

20.1.3 随机梯度下降法

这让我们想到了做近似梯度下降的问题，或 "随机 "梯度下降。把成本景观看作是一个李亚普诺夫函数†，那么任何每一步都在下坡的更新最终都会达到最优。更一般地说，任何平均下坡的更新最终都会达到最小值.....有时 "随机 "梯度下降更新偶尔上坡，但平均下坡，甚至可以有理想的特性，如反弹出小的局部最小值。下面的图给出了一些图形上的直觉；关于随机梯度方法的正式处理，见例如[5].

†的李亚普诺夫函数$V(\alpha)=g(\alpha)-g(\alpha^*)$，通常用于优化算法的收敛/稳定性分析。

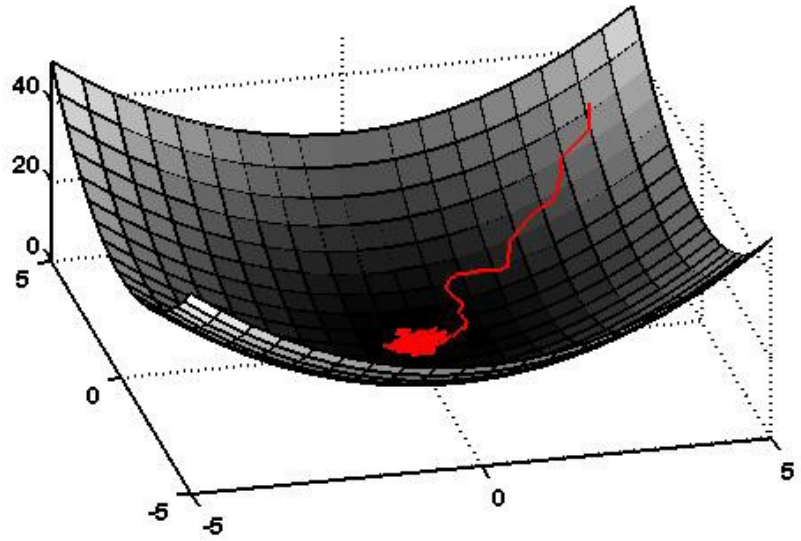


图 -20.1 二次方成本函数的随机梯度下降。

20.1.4 权重推理算法

与其对每个维度独立采样，不如考虑对参数向量α做一个小的随机变化，即β。现在考虑更新的形式。

$$\Delta\alpha=-\eta[g(\alpha+\beta)-g(\alpha)]\beta。$$

这里的直觉是，如果$g(\alpha+\beta)<g(\alpha)$，那么$\beta$是一个很好的变化，我们会向$\beta$的方向移动；如果成本增加，那么我们会向相反方向移动。假设函数是平滑的，β是小的，那么我们将永远在李亚普诺夫函数上下坡移动。

更有趣的是，平均而言，更新实际上是朝着真实梯度的方向进行的。为了看到这一点，我们可以再次假设该函数是局部平滑的，并且β很小，以写出泰勒展开。

$$g(\alpha + \beta) \approx g(\alpha) + \frac{\partial g}{\partial \alpha}\beta$$

现在我们有

$$\Delta\alpha \approx -\eta\left[\frac{\partial g}{\partial \alpha}\beta\right]\beta = -\eta\beta\beta^T\frac{\partial g}{\partial \alpha}^T$$

$$E[\Delta\alpha] \approx -\eta E[\beta\beta^T]\frac{\partial g}{\partial \alpha}^T$$

如果我们从一个均值为零、方差为σ^2的分布中独立选择β的每个元素，或者说$E[\beta_i] = 0$，$E[\beta_i\beta_j] = \sigma^2\delta_{ij}$，那么我们就有$\beta\beta$

$$E[\Delta\alpha] \approx -\eta\sigma_\beta^2\frac{\partial g}{\partial \alpha}^T。$$

请注意，分布$p_\alpha(\mathbf{x})$不一定是高斯的，但正是分布的方差决定了梯度上的缩放。

20.1.5 用估计的基线进行重量扰动

上面的权重扰动更新要求我们在每次更新参数时评估函数g两次。这与Ønite diÆerences方法相比是很低的，但让我们看看我们是否可以做得更好。如果我们不在每次更新时评估函数[$g(\alpha+\beta)$和$g(\alpha)$]，而是用从以前的试验中得到的估计值$b=\hat{g}(\alpha)$来代替$g(\alpha)$的评估呢？ 换句话说，考虑一个更新的形式。

$$\Delta\alpha = -\frac{\eta}{\sigma_\beta^2}[g(\alpha + \beta) - b]\beta \tag{2}$$

估计器可以采取多种形式，但最简单的也许是基于更新（第n次试验后）。

$$b[n + 1] = \gamma g[n] + (1 - \gamma)b[n], \quad b[0] = 0, 0 \le \gamma \le 1,$$

其中γ为移动平均数的参数。让我们计算一下更新的期望值，使用与上面相同的泰勒扩展。

$$\begin{aligned} E[\Delta\alpha] &= -\frac{\eta}{\sigma_\beta^2}E[[g(\alpha) + \frac{\partial g}{\partial \alpha}\beta - b]\beta]。 \\ &= -\frac{\eta}{\sigma^2}E[[g(\alpha) - b]\beta] - \frac{\eta}{\sigma^2}E[\beta\beta]\frac{\partial g}{\partial \alpha}^T \\ &= -\eta\frac{\partial g}{\partial \alpha}^T \end{aligned}$$

换句话说，基线并不影响我们的基本结果--预期的更新是在梯度的方向上。请注意，这种计算对任何与β不相关的基线估计器都是有效的，如果估计器是以前试验中的性能的函数，就应该如此。

尽管使用估计的基线不会影响平均更新，但它会对算法的性能产生巨大的影响。正如我们在后面的章节中所看到的，如果对g的评价是随机的，那么使用基线估计的更新实际上可以胜过使用直接函数评价的更新。

让我们来看看b=0.的极端情况，这似乎是一个非常糟糕的想法.....在每一步中，我们将向每个随机扰动的方向移动，但我们将根据评估的成本在这个方向上或多或少地移动。平均来说，我们仍然会向真正的梯度方向移动，但只是因为我们最终会向下坡移动的次数多于上坡。这感觉非常天真。

20.1.6 REINFORCE w/ additive Gaussian noise

现在让我们考虑REINFORCE更新的简单形式。

$$\frac{\partial}{\partial\alpha}E\left[g(\mathbf{x})\right] = E\left[g(\mathbf{x})\,\frac{\partial}{\partial\alpha}\log p_\alpha(\mathbf{x})\right]。$$

事实证明，权重扰动是一种REINFORCE算法。要看到这一点，请拿$\mathbf{x}=\alpha+\beta$，$\beta\in N$（0，σ^2），得

$$p_\alpha(\mathbf{x})=\frac{1}{(2\pi\sigma)^{N/2}}e^{-(\mathbf{x}-\alpha)^T(\mathbf{x}-\alpha)/2\sigma^2}$$

$$\log p(_\alpha\mathbf{x}) = \frac{-(\mathbf{x}-\alpha)^T(\mathbf{x}-\alpha)}{2\sigma^2} + 不依赖于\alpha的条款$$

$$\frac{\partial}{\partial\alpha}\log p_{\alpha}(\mathbf{x}) = \frac{1}{2}(\alpha-\mathbf{x})^T = \frac{1}{\sigma\sigma^2}\beta^T。$$

如果我们在每次蒙特卡洛评估中只使用一个试验，那么REINFORCE更新为

$$\Delta\alpha = -\frac{\eta}{\sigma^2}g(\alpha+\beta)\beta。$$

这正是权重扰动更新（上面讨论的$b=0$的疯狂版本）。虽然它平均会朝着梯度的方向移动，但它可能是高度不Æicient的。在实践中，人们使用了许多不止一个样本来估计政策梯度。

20.1.7 摘要

REINFORCE中使用对数概率的政策梯度 "技巧 "提供了一种估计真实政策梯度的方法。这不是获得政策梯度的唯一方法.....事实上，对于均值为α线性、对角线协方差为Øxed的政策，琐碎的权重扰动更新可以获得相同的梯度。它的聪明之处在于，它利用了我们所拥有的信息（瞬时成本值和政策的梯度），并且它提供了一个无偏的梯度估计（注意，取一个只有轻微错误的模型的梯度可能不具有这种优点）。但它的不公正来自于有一个潜在的非常高的方差。减少政策梯度的方差，通常是通过基线估计，仍然是一个活跃的研究领域。

20.2 通过信噪比来衡量样本性能。

20.2.1 权重扰动的性能

REINFORCE/权重扰动更新的简单性使得它很有可能被应用于任意复杂度的问题。但该算法的一个主要问题是它的性能--尽管我们已经表明，更新*平均是朝着*真实梯度的方向进行的，但它可能仍然需要大量的计算来获得局部最小值。

在本节中，我们将通过调查权重扰动算法的信噪比（SNR）来研究该算法的性能。这一想法在以下方面进行了探索

[1]-- 我的目标只是在这里给你一个提示。信噪比是信号中的功率（这里是在真实梯度方向上的期望更新）和噪声中的功率（更新的剩余部分，因此$\Delta\alpha=-\eta^T\frac{\partial}{\partial\alpha}$+噪声）的比率。 信噪比也可以被定义为

$$\text{SNR} = \frac{-\eta \left(\frac{\partial g}{\partial \alpha}\right)^T \left(\frac{\partial g}{\partial \alpha}\right)^T}{E\left[\Delta\alpha + \eta \frac{\partial g}{\partial \alpha}\right]}.$$

在无偏更新的特殊情况下，该方程简化为。

$$\text{SNR} = \frac{E[\Delta\alpha]^T E[\Delta\alpha]}{E\left[(\Delta\alpha)^T(\Delta\alpha)\right] - E[\Delta\alpha]^T E[\Delta\alpha]}。$$

对于权重扰动的更新，我们有。

$$E[\Delta\alpha]^T E[\Delta\alpha] = \eta^2 \frac{\partial g}{\partial \alpha}^T \frac{\partial g}{\partial \alpha} = \eta^2 \sum_{i=1}^{N} \left(\frac{\partial g}{\partial \alpha_i}\right)^2,$$

$$E\left[(\Delta\alpha)^T(\Delta\alpha)\right] = \frac{\eta^2}{\sigma_\beta^4} E\left[(g(\alpha+\beta)-g(\alpha))^2 \beta^T \beta\right]。$$

$$\approx \frac{\eta^2}{\sigma_\beta^4} E\left[\left(\frac{\partial g}{\partial \alpha}\beta\right)^2 \beta^T \beta\right]。$$

$$= \frac{\eta^2}{\sigma_\beta^4} E\left[\left(\sum_i \frac{\partial g}{\partial \alpha_i}\beta_i\right)\left(\sum_j \frac{\partial g}{\partial \alpha_j}\beta_j\right)\left(\sum_k \beta_k^2\right)\right]$$

$$= \frac{\eta^2}{\sigma_\beta^4} E\left[\sum_{i,j,k} \frac{\partial g}{\partial \alpha_i}\frac{\partial g}{\partial \alpha_j}\beta_i \beta_j \beta_k^2\right]$$

$$= \frac{\eta^2}{\sigma_\beta^4} \sum_{i,j,k} \frac{\partial g}{\partial \alpha_i}\frac{\partial g}{\partial \alpha_j} E\left[\beta_i \beta_j \beta_k^2\right]$$

$$E\left[\beta_i \beta_j \beta_k^2\right] = \begin{cases} 0 & i \neq j \\ \sigma_\beta^4 & i = j \neq k \\ \mu_4(\beta) & i = j = k \end{cases}$$

$$= \eta^2 (N-1) \sum_i \left(\frac{\partial g}{\partial \alpha_i}\right)^2 + \frac{\eta^2 \mu_4(z)}{\sigma_\beta^4} \sum_i \left(\frac{\partial g}{\partial \alpha_i}\right)^2$$

其中$\mu_n(z)$是z的第n个中心矩。

$$\mu_n(z) = E\left[(z - E[z])^n\right]。$$

把这一切放在一起，我们有。

$$\text{信噪比} = \frac{1}{N - 2 + \frac{\mu_4(\beta_i)}{\sigma_\beta^4}}。$$

例子20.1(附加高斯噪声的信噪比)

对于从高斯分布中抽取的β_i，我们有$\mu_1 = \mu_{0,2} = \sigma_\beta^2$，$\mu_3 = \mu_{0,4} = 3\sigma_\beta^4$。将上述表达式简化为。

$$\text{SNR} = \frac{1}{N+1}.$$

例子20.2（附加均匀噪声的信噪比）。

对于 θ_i 在[-a,a]上 $\mu_1 = \mu_0, \mu_2 = \frac{a^2}{3} = \sigma_\theta^2, \mu_3 = \mu_0, \mu_4 = \frac{a^4}{5} = \frac{9}{5}\sigma_\theta^4$ 的均匀分布，我们有特上述简化为。

$$\text{SNR} = \frac{1}{N\frac{1}{5}}.$$

使用信噪比的性能计算可以用来设计实践中的算法参数。例如，基于这些结果，很明显，通过均匀分布添加的噪声在非常小的N的情况下比高斯噪声产生更好的梯度估计，但这些差异在大N的情况下是可以忽略的。

类似的计算可以产生对挑选加性噪声的大小（以σ_θ为尺度）的洞察力。本节的计算似乎暗示，更大的σ_θ只能减少方差，克服基线估计器中的误差或噪声，$\tilde{b}$；这是一个我们的一阶泰勒扩展的缺点。如果成本函数在参数中不是线性的，那么对高阶项的检查显示，大的σ_θ可以增加信噪比。二阶泰勒扩展的推导留作练习。

参考文献

1. 约翰-W-罗伯茨，"通过改进的{SNR}在翻转板上的运动学习"。算法"，，2月，[2009.链接]。
2. Peter W. Glynn, "Likelihood ratio gradient estimation for stochastic systems", *Communications of the ACM*, vol. 33, no. 10, pp.75-84, oct, 1990.
3. R.J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning", *Machine Learning*, vol. pp8,.229-256, 1992.
4. William H. Press and Saul A. Teukolsky and William T. Vetterling and Brian P. Flannery, "Numerical Recipes in C。科学计算的艺术"，剑桥大学出版社。1992.
5. Dimitri P Bertsekas, "Nonlinear programming",Athena scientiØc Belmont, 1999.

低动能机器人技术

走路、跑步、游泳、飞行和操纵的算法

吕斯-特德雷克

最后修改 2021-6-13.

注意：这些是用于在麻省理工学院教授的课程的工作笔记。它们将在2021年春季学期中被更新。讲座视频可在YouTube上找到。

上一章 目录 下一章

附录A
德雷克

DRAKE是我们在这个文本中使用的主要软件工具箱，事实上，它的起源在很大程度上是由于麻省理工学院的欠发达课程。DRAKE网站是信息和文档的主要来源。本章的目的是提供任何额外的信息，以帮助你启动和运行这些笔记中提供的例子和练习。

A.1 PYDRAKE

DRAKE主要是一个C++库，具有严格的编码标准和成熟度，甚至可以支持工业领域的专业应用。为了提供一个更温和的介绍，并促进快速的原型设计，我完全用python写这些笔记，使用Drake的python绑定（pydrake）。这些绑定没有C++后端那么成熟；我们非常欢迎你的反馈（甚至是贡献）。它仍在迅速改进。

特别是，虽然C++的API文档非常好，但自动生成的Python文档仍然是一个进展中的工作。我目前建议使用C++文档来寻找你需要的东西，然后只有在你需要了解类或方法在pydrake中是如何拼写的时候才查看Python文档。

DRAKE中也有一些教程，可以帮助你入门。

A.2 在线jupyter笔记本

我将以Jupyter笔记本的形式提供几乎所有的例子和练习，这样我们就可以利用奇妙的、相对较新的（免费）云资源。特别是我选择集中使用谷歌的Colaboratory（Colab）作为我们的主要平台。大多数章节都有一个相应的笔记本，你可以在云上运行，而不需要在你的机器上安装任何东西！

Colab是一个非常棒的资源，拥有慷慨的计算资源和一个非常方便的界面，允许你在Google Drive上保存/加载/分享你的工作。虽然我们不能提前提供机器，但我们可以

提供它们

在每台笔记本的顶部都有一个略显丑陋的安装脚本。当你第一次打开每台笔记本时，需要大约两分钟来配置机器。在写这篇文章的时候，如果你关闭浏览器，似乎机器将为你保持配置90分钟，如果你保持开放，则为12小时。配置是按用户进行的，似乎是相对无限的。

A.3 在你自己的机器上运行

随着你越来越高级，你可能会想在自己的机器上运行（和扩展）这些例子。DRAKE网站有许多安装选项，包括预编译的二进制文件和Docker实例。在这里，我提供了一个从预编译的二进制文件设置Drake和运行教科书中的例子的工作流程。

首先，选择你的平台（点击你的操作系统）。

Ubuntu Linux (Bionic) | Mac Homebrew

A.3.1 安装德雷克

下面的链接显示了本文中的例子所测试的特定分布。

下载二进制文件

```
curl -o drake.tar.gz https://drake-packages.csail.mit.edu/drake/nightly/drake-latest-bionic
```

解压并设置你的PYTHONPATH和测试

```
sudo tar -xvzf drake.tar.gz -C /opt
sudo /opt/drake/share/drake/setup/install_prereqs
export PYTHONPATH=/opt/drake/lib/python3.6/site-packages:${PYTHONPATH}
```

A.3.2 下载教科书补编

```
git clone https://github.com/RussTedrake/underactuated.git
```

并使用提供的平台专用安装脚本安装先决条件。

注意，像往常一样，我强烈建议在virtualenv中运行pip命令。

```
cd的作用力不足
sudo scripts18.04/setup/ubuntu//install_prereqs.sh
pip3 install --requirement requirements.txt
export PYTHONPATH=`pwd`:${PYTHONPATH}.
```

A.3.3 运行Jupyter笔记本

你很可能想从未激活的根目录开始。然后用以下方法启动你的笔记本

```
jupyter笔记本
```

每一章的例子都会在一个.ipynb Øle中，就在该章的html Øle旁边，而笔记本的练习都位于练习的子目录中。

A.4 获得帮助

如果你在使用DRAKE时遇到麻烦，请遵循这里的建议。如果你在使用未激活的 repo 时遇到麻烦，你可以在这里检查已知的问题（并有可能Øle一个新的问题）。

低动能机器人技术

走路、跑步、游泳、飞行和操纵的算法

吕斯-特德雷克

最后修改 2021-6-13.

注意：这些是用于在麻省理工学院教授的课程的工作笔记。它们将在2021年春季学期中被更新。讲座视频可在YouTube上找到。

上一章　　目录　　下一章

附录B
多体动力学

B.1 推导运动方程

一个标准机器人的运动方程可以用拉格朗日的方法推导出来。 用T作为系统的总动能，U作为系统的总势能，$L = T - U$，Q_i作为对应于q的$_i$广义力，拉格朗日动态方程为。

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = Q_i. \tag{1}$$

你可以把它们看作是牛顿方程的一般化。对于一个粒子来说。我们有$T = \frac{1}{2}m\dot{x}^2$，所以$\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} = m\ddot{x}$，并且$\frac{\partial L}{\partial x} = -\frac{\partial U}{\partial x}$。$-\frac{\partial U}{\partial x} = f$，相当于$f = ma$。但拉格朗日推导法在广义坐标系和受限运动中起作用。

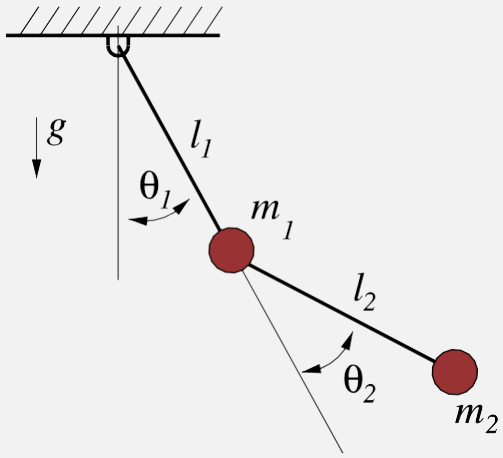
不谈全部细节，处理约束的关键思想是 "虚功原理"（以及相关的达姆贝尔原理）。为了用牛顿方法描述一个处于平衡状态的钟摆，我们必须计算外力（如重力）和内力（如保持钟摆臂与桌面相连的力），并使它们的总和等于零。计算内力对摆来说稍微有点麻烦；而对更复杂的机构来说，这就变得难以忍受了。如果力的总和等于零，那么这些力在一些（不确定的）虚拟位移中所做的功肯定也等于零。现在的诀窍是：如果我们只考虑与运动学约束一致的虚拟位移（例如，围绕钟摆的针状关节的旋转），那么我们可以计算虚拟功并将其设为零，而无需计算内力将这一想法扩展到动力学的情况下（通过达姆贝尔和汉密尔顿原理），最终形成上述的拉格朗日方程。

如果你对这些方程不适应，那么任何一本关于刚体力学的好书的章节都可以让你赶上进度。[1]是一本关于机器人运动学/动力学的优秀实用指南。但是[2] 是我迄今为止最喜欢的力学书；如果你想了解更多，我强烈推荐它。现在继续把拉格朗日力学简单地看作是一个配方，你可以把它应用于一个伟大的项目中，这也是可以的。

许多情况下，产生运动方程。

为了完整起见，我在下面附上了从静止作用原理推导出的拉格朗日。

例B.1(简单双摆)



图B.1 - 简单双摆

考虑简单的双摆，在两个关节处都有扭矩驱动，所有的质量都集中在两点（为简单起见）。 用$\mathbf{q}=[\vartheta_1, \vartheta]_2^T$，$\mathbf{p}_1$，$\mathbf{p}_2$分别表示$m_1$，$m$的$_2$位置，这个系统的运动学是。

$$\mathbf{p} = {}_1 l_1 \begin{bmatrix} s1 \\ -c_1 \end{bmatrix}, \mathbf{p}_2 = \mathbf{p}_1 + l_2 \begin{bmatrix} s_{1+2} \\ -c_{1+2} \end{bmatrix}$$

$$\dot{\mathbf{p}} = {}_{11}\dot{q}_1 \begin{bmatrix} c_1 \\ s_1 \end{bmatrix}, \dot{\mathbf{p}}_2 = \dot{\mathbf{p}}_1 + l({}_2\dot{q}_1 + \dot{q}_2) \begin{bmatrix} c_{1+2} \\ s1+2 \end{bmatrix}$$

请注意，s_1是sin(q_1)的简写，c_{1+2}是cos(q_1+q)的$_2$简写，等等。由此可见我们可以写出动能和势能。

$$T = \frac{1}{2} m_1 \dot{\mathbf{p}}_1^T \dot{\mathbf{p}}_1 + \frac{1}{2} m_2 \dot{\mathbf{p}}_2^T \dot{\mathbf{p}}_2$$

$$= \frac{1}{2}(m_1 + m_2) {}_2 l_1^2 \dot{q}_1^2 + \frac{1}{2} m_2 l_1^2 (\dot{q}_2 + \dot{q}_2)^2 + m_2 l_1 l_2 \dot{q}_1 (\dot{q}_1 + \dot{q}_2) c$$

$$U = {}_1 mgy_1 + {}_2 mgy_2 = -(m_1 + m_2) {}_1 glc_1 - {}_2 mglc_{21+2}$$

取部分导数$\frac{\partial T}{\partial \dot{q}_i}$，，$\frac{\partial T}{\partial q_i}$和（$\frac{\partial U}{\partial q_i}$项总是零），则$\frac{d}{dt}\frac{\partial T}{\partial \dot{q}_i}$将它们插入拉格朗日，就可以看到运动方程。

$$(m_1 + m) {}_2 l_1^2 \ddot{q}_1 + {}_2 ml_1^2 (\ddot{q}_1 + \ddot{q}_2) + {}_{21} mll_2 (2\ddot{q}_1 + \ddot{q})_2 c_2$$

$$- mll_{21} (_2 2\dot{q}_1 + \dot{q})_2 \dot{q}_{22} s + (m_1 + m_2) lgs_{11} + {}_2 mgls_{21+2} =$$

$${}_2 ml_1^2 (\ddot{q}_1 + \ddot{q}_2) + ml_{2121} c_2 + ml_{212}^2 s_2 + {}_{12} mgls_{21+2} = \tau_2$$

正如我们在本章中看到的，在DRAKE中对这些方程进行数值积分1,（和动画）会产生预期的结果。

B.2 操纵器方程

如果你通过对几个简单的机器人操纵器的拉格朗日动力学进行曲解，你会开始看到一个模式的出现--所产生的运动方程都有一个特征形式。例如，你的机器人的动能总是可以是

写在表格里。

$$T = \frac{1}{2}\dot{\mathbf{q}}^T \mathbf{M}(\mathbf{q})\dot{\mathbf{q}}。 \tag{2}$$

其中$\mathbf{M}$是依赖于状态的惯性矩阵（又称质量矩阵）。这一观察对一般的操纵器动力学有一些启发--例如，我们知道$\mathbf{M}$总是正的非线性，而且是对称的[...]。3，第107页]，并且有一个美丽的稀疏模式[.4]，我们应该在我们的算法中加以利用。

继续我们的抽象，我们发现一般机器人操纵器的运动方程（没有运动学循环）的形式为

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} = \tau(_g\mathbf{q}) + \mathbf{Bu}, \tag{3}$$

其中$\mathbf{q}$是关节位置矢量，$\mathbf{M}$是惯性矩阵，$\mathbf{C}$捕捉科里奥利力，τ_g是重力矢量。矩阵$\mathbf{B}$将控制输入$\mathbf{u}$映射到
广义的力量。 请注意，我们在左边配对$\mathbf{M}\ddot{\mathbf{q}}$+$\mathbf{C}\dot{\mathbf{q}}$是因为"..。
运动方程取决于坐标$\mathbf{q}$的选择。由于这个原因，$\mathbf{M}\ddot{\mathbf{q}}$和$\mathbf{C}\dot{\mathbf{q}}$都不应该被认为是一个广义的力；只有它们的总和才是一个力"[5](s.10.2).这些项代表动能对拉格朗日的贡献。

$$\frac{d}{dt}\frac{\partial T^T}{\partial \dot{\mathbf{q}}} - \frac{\partial T^T}{\partial \mathbf{q}} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}。$$

每当我写出公式(3)，我就看到$ma = f$。

例B.2（简单双摆的操纵器方程式形式）。

例1的运动方程可以紧凑地写成：。

$$\mathbf{M}(\mathbf{q}) = \begin{bmatrix} (m_1 + m_2)l_1^2 + m_2 l_2^2 + 2m_2 l_1 l_2 c_2 & m_2 l_2^2 + m_2 l_1 l_2 c_2 \\ m_2 l_2^2 + m_2 l_1 l_2 c_2 & m_2 l_2^2 \end{bmatrix}$$

$$\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \begin{bmatrix} 0 & -m_2 l_1 l_2 (2\dot{q}_2 + \dot{q}_1) s_2 \\ m_2 l_1 l_2 s_2 \dot{q}_1 & 0 \end{bmatrix}$$

$$\tau_g(\mathbf{q}) = -g \begin{bmatrix} (m_1 + m_2) l_1 s_1 + m_2 l_2 s_{1+2} \\ m_2 l_2 s_{1+2} \end{bmatrix}。 \qquad \mathbf{B} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

请注意，$\mathbf{C}$矩阵的这种选择不是唯一的。

操纵器方程是非常一般的，但它们确实定义了一些重要的特征。 例如，$\ddot{\mathbf{q}}$与控制输入$\mathbf{u}$呈线性关系（与状态有关）。这一观察证明了整个说明中假设的动力学的控制形式是合理的。还有许多其他重要的属性可能被证明是有用的。例如，我们知道$Cq^{\cdot}$的i元素由[3, §5.2.3]:

$$[\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}]_i = \sum_{j,k} c_{ijk}(\mathbf{q})\dot{q}_j\dot{q}_k, \qquad c_{ijk}(\mathbf{q}) = \frac{\partial M_{ij}}{\partial q_k} - \frac{1}{2}\frac{\partial M_{jk}}{\partial q_i}。$$

(例如，它也是对$\dot{\mathbf{q}}$的二次方)，对于$\mathbf{C}$的适当选择，我们有
$\mathbf{M}(\mathbf{q})$-2$\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$是偏斜对称的[6].

请注意，我们选择在本书中使用二阶系统的符号（在方程中出现$\dot{\mathbf{q}}$和$\ddot{\mathbf{q}}$）。 尽管我认为它提供了更多的清晰度，但这种符号有一个重要的限制：使用这种符号不可能在不引入运动学奇异性（如著名的 "万向节锁"）的情况下以 "最小坐标 "描述三维旋转。例如，代表三维旋转的一个常见的无奇异性选择是单位四元数，由4个实值描述（加上一个规范约束）。然而，我们仍然可以在没有奇异性的情况下，只用实值3来表示旋转速度。这意味着

我们的速度向量的长度不再与我们的位置向量的长度相同。由于这个原因，你会看到DRAKE中的大多数软件都使用了更通用的符号，用**v**表示速度，用**q**表示位置，操纵器方程写为

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} + \mathbf{C}(\mathbf{q}, \mathbf{v})\mathbf{v} = \tau_g(\mathbf{q}) + \mathbf{B}\mathbf{u}, \tag{4}$$

这不一定是一个二阶系统。见[7]对这一主题进行了很好的讨论。

B.2.1 递归动力学算法

我们的机器的运动方程很快就会变得复杂。幸运的是，对于具有树状链接运动学结构的机器人，有非常eÆicient和自然的递归算法来生成这些运动方程。关于这些方法的详细参考资料，见[8]；有些人更喜欢阅读[]中的铰接体方法。9].DRAKE中的实现使用了[ ]中的相关表述。10].

B.2.2 汉密尔顿力学

在某些情况下，最好使用动力学的哈密顿公式，用状态变量**q**和**p**（而不是$\dot{\mathbf{q}}$），其中**p** $= = \frac{\partial L^T}{\partial \dot{\mathbf{q}}}\mathbf{M}\dot{\mathbf{q}}$的结果为

的动态。

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{q}} = \mathbf{p}$$

$$\dot{\mathbf{p}} = \mathbf{c}_H(\mathbf{q}, \mathbf{p}) + \tau_g(\mathbf{q}) + \mathbf{B}\mathbf{u}, \qquad \text{其中，} c\ (q_H,\ \mathbf{p}) = \left(\frac{1}{2}\frac{\partial[\dot{\mathbf{q}}^T\mathbf{M}(\mathbf{q})\dot{\mathbf{q}}]}{\partial \mathbf{q}}\right)^T.$$

回想一下，汉密尔顿是$H = \frac{\partial}{\partial \mathbf{p}}{}^{pT}\dot{\mathbf{q}} - L$；这些运动方程是我们熟悉的$\dot{\mathbf{q}} =$ ，$\frac{\partial H^T}{\partial}\dot{\mathbf{p}} = \frac{\partial H}{\partial \mathbf{q}}{}^T$。

B.2.3 双边立场的限制

如果我们的机器人有闭合的运动学链，例如那些由四杆联动产生的运动学链，那么我们需要多一点。上面的拉格朗日机制假设了 "最小坐标"；如果我们的状态向量**q**包含了运动学链中的所有环节，那么我们就没有最小参数化--每个运动学回路都增加了（至少）一个约束，所以应该删除（至少）一个自由度。尽管有些约束可以被解决掉，但更普遍的解决方案是使用拉格朗日来推导无约束系统的动力学（没有闭环约束的运动学树），然后添加额外的广义力，确保约束总是被满足。

考虑一下约束方程

$$\mathbf{h}(\mathbf{q}) = 0. \tag{5}$$

对于运动学闭合链的情况，这可以是运动学的约束，即链的一端的位置等于链的另一端的位置。运动方程可以写成

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} = \tau_g(\mathbf{q}) + \mathbf{B}\mathbf{u} + \mathbf{H}^T(\mathbf{q})\lambda, \tag{6}$$

其中$\mathbf{H}(\mathbf{q}) = \frac{\partial \mathbf{h}}{\partial \mathbf{q}}$和$\lambda$是约束力。让我们使用速记的方式

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} = \tau(q, \dot{q}, u) + \mathbf{H}^T(\mathbf{q})\lambda. \tag{7}$$

使用

$$\dot{\mathbf{h}} = \mathbf{H}\dot{\mathbf{q}}。 \tag{8}$$

$$\ddot{\mathbf{h}} = \mathbf{H}\ddot{\mathbf{q}} + \dot{\mathbf{H}}\dot{\mathbf{q}} \tag{9}$$

。

我们可以求解$\boldsymbol{\lambda}$，通过观察，当约束被施加时，$\mathbf{h}=0$，因此$\dot{\mathbf{h}}=$和$0\ddot{\mathbf{h}}=0$。 将动态(7)插入(9)得到

$$\boldsymbol{\lambda}=-(^{-1}\mathbf{HMH}^{T})^{+}(\mathbf{HM}^{-1}\boldsymbol{\tau}+\dot{\mathbf{H}}\dot{\mathbf{q}})。 \tag{10}$$

该+符号指的是摩尔-彭罗斯的伪逆。在许多情况下，这个矩阵将是全等级的（例如，在多个独立的四杆连杆的情况下），可以使用传统的逆。当矩阵下降等级时（多个解决方案），那么伪逆将选择在最小平方意义上约束力最小的解决方案。

为了防止数字上的 "约束漂移"，人们可能希望在约束没有满足数字精度的情况下增加一个恢复力。为了达到这个目的，我们可以不像0上面那样求解$\ddot{\mathbf{h}}=$，而是求解为

$$\ddot{\mathbf{h}} = \mathbf{H}\ddot{\mathbf{q}} + \dot{\mathbf{H}}\dot{\mathbf{q}} = -\alpha 2\dot{\mathbf{h}} - \alpha^{2}\mathbf{h}。 \tag{11}$$

其中$\alpha>0$是一个刚性参数。这被称为Baumgarte的稳定技术，在这里用一个参数来实现，以提供一个临界阻尼响应。贯彻这一点，可以得到

$$\boldsymbol{\lambda} = -(^{-1}\mathbf{HMH}^{T})^{+}(\mathbf{HM}^{-1}\boldsymbol{\tau} + (\dot{\mathbf{H}} + 2\alpha\mathbf{H})\dot{\mathbf{q}} + \alpha^{2}\mathbf{h})。 \tag{12}$$

这些方程有一个重要的基于优化的推导/解释，我们将在下面使用高斯的最小约束原则来建立这个解释。这个原理说，我们可以推导出受约束的动力学为 。

$$\begin{aligned} &\text{闽} \quad \frac{1}{2}(\ddot{\mathbf{q}}-\ddot{\mathbf{q}})_{uc})^{T}\mathbf{M}(\ddot{\mathbf{q}}-\ddot{\mathbf{q}})_{uc}), \\ &\text{受制于} \quad \ddot{\mathbf{h}}(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) = = 0\mathbf{H}\ddot{\mathbf{q}} + \dot{\mathbf{H}}\dot{\mathbf{q}}。 \end{aligned} \tag{13}$$

其中$\ddot{\mathbf{q}}_{uc}=\mathbf{M}^{-1}\boldsymbol{\tau}$是 "无约束的加速度"[...11].方程（6）直接来自于优化条件，而$\boldsymbol{\lambda}$正好作为拉格朗日乘数。这是一个带有平等约束的凸二次方程，它有一个封闭式的解决方案，正是上面的方程（10）。看一下这个优化的对偶表述也很有意义，它可以写为

$$\min_{\lambda}\frac{1}{2}\boldsymbol{\lambda}^{T-1}\mathbf{HMH}^{T}\boldsymbol{\lambda} - \boldsymbol{\lambda}^{T}（\mathbf{HM}^{-1}\boldsymbol{\tau} + \dot{\mathbf{H}}\dot{\mathbf{q}}）。$$

请注意，线性项包含$\mathbf{h}$的加速度，如果动力学演化时没有施加约束力；'让我们称其为$\ddot{\mathbf{h}}_{uc}$。

$$\min_{\boldsymbol{\lambda}}\frac{1}{2}\boldsymbol{\lambda}^{T-1}\mathbf{HMH}^{T}\boldsymbol{\lambda} - \boldsymbol{\lambda}^{T}\ddot{\mathbf{h}}_{uc}。$$

原始公式将加速度作为决策变量，而双重公式将约束力作为决策变量。现在，这只是一个可爱的观察；当我们讨论接触力时，我们会在此基础上更进一步。

B.2.4 双边速度限制

考虑一下约束方程

$$\mathbf{h}_{v}(\mathbf{q},\dot{\mathbf{q}}) = 0 , \tag{14}$$

其中$\frac{\partial \mathbf{h}_{eq}}{\partial \dot{\mathbf{q}}} \neq$ 这些0.不太常见，但当例如一个关节通过规定的运动被驱动时就会出现。在这里，操纵器方程由以下几项给出

$$\mathbf{M}\ddot{\mathbf{q}} = \tau + \frac{\partial \mathbf{h}_v}{\partial \dot{\mathbf{q}}}^T \lambda \quad .(15)$$

使用

$$\dot{\mathbf{h}}_v = \frac{\partial h_v}{\partial \mathbf{q}} \dot{\mathbf{q}} + \frac{\partial h_v}{\partial \dot{\mathbf{q}}} \ddot{\mathbf{q}}。 \quad (16)$$

设$\dot{\mathbf{h}}_v = 0$yield

$$\lambda = -\left(\frac{\partial \mathbf{h}_v}{\partial \dot{\mathbf{q}}} \mathbf{M}^{-1} \frac{\partial \mathbf{h}_v}{\partial \dot{\mathbf{q}}}\right)^+ \left[\frac{\partial h_v}{\partial \dot{\mathbf{q}}} \mathbf{M}^{-1} \tau + \frac{\partial h_v}{\partial \mathbf{q}} \dot{\mathbf{q}}\right] \quad (17)$$

同样，为了防止约束漂移，我们可以要求$\dot{\mathbf{h}}_v = -\alpha \mathbf{h}_v$，这样就得到了

$$\lambda = -\left(\frac{\partial \mathbf{h}_v}{\partial \dot{\mathbf{q}}} \mathbf{M}^{-1} \frac{\partial \mathbf{h}_v}{\partial \dot{\mathbf{q}}}\right)^+ \left[\frac{\partial h_v}{\partial \dot{\mathbf{q}}} \mathbf{M}^{-1} \tau + \frac{\partial h_v}{\partial \mathbf{q}} \dot{\mathbf{q}} + \alpha \mathbf{h}_v\right]。 \quad (18)$$

B.3 接触的动力

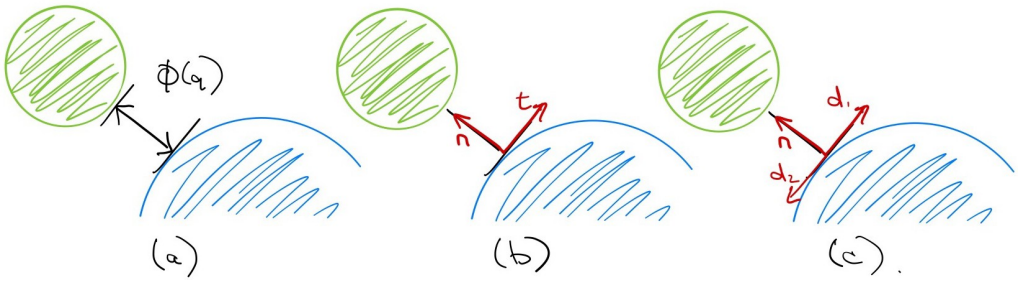
进行和中断接触的多体系统的动力学与受约束系统的动力学密切相关，但往往要复杂得多。在最简单的形式下，你可以把非穿透看作是一个*不等式*约束：碰撞体之间的签名距离必须是非负的。但是，正如我们在关于行走的章节中所看到的，当这些约束变得活跃时，其转换对应于碰撞，对于有动量的系统，它们需要一些小心。我们还将看到，摩擦性接触增加了它自己的挑战。

有三种主要方法用于对 "刚性 "体系统的接触进行建模。1）刚性接触近似于顺应性接触；2）带有碰撞事件检测、冲动复位图和碰撞事件之间的连续（约束）动力学的混合模型；3）刚性接触近似于时间步进方案中的时间平均力（脉冲）。每种建模方法对于不同的应用都有优势/劣势。

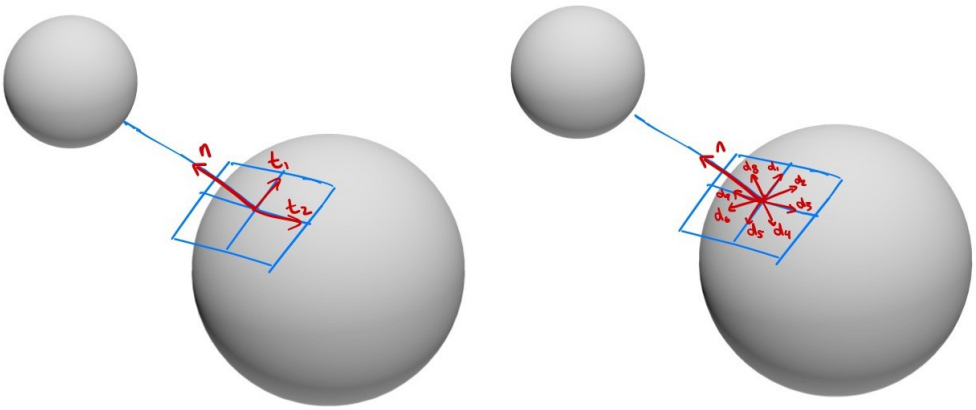
在我们开始之前，有一点我们将全程使用的符号。让$j(\mathbf{q})$表示两个刚性体之间的相对（有符号）距离。对于刚体接触，我们希望执行单边约束。

$$j(\mathbf{q}) \geq 0. \quad (19)$$

我们将使用$\mathbf{n} = \frac{\partial \phi}{\partial \mathbf{q}}$来表示 "接触法线"，以及$\mathbf{t}_1$和$\mathbf{t}_2$作为一个基础的触点处的切线（笛卡尔空间的正交矢量，投射到关节空间），都以关节坐标的行矢量表示。



图B.2 - 二维的接触坐标。(a) 接触体之间的符号距离，$j(\mathbf{q})$。(b) 法线$(\mathbf{n})$和切线$(\mathbf{t})$接触矢量--注意，当$\mathbf{q}$是其中一个体的x、y位置时，这些矢量可以在二维中画出来，但更普遍的是这些矢量存在于构架空间中。 (c) 有时用d_1和d来$_2$表达切向坐标对我们有帮助；这在三维情况下会更有意义。



图B.3 - 3D中的接触坐标。

我们还将发现，将接触法线和切线向量组合成一个单一的矩阵**J**，我们称之为*接触雅各布系数*，是非常有用的。

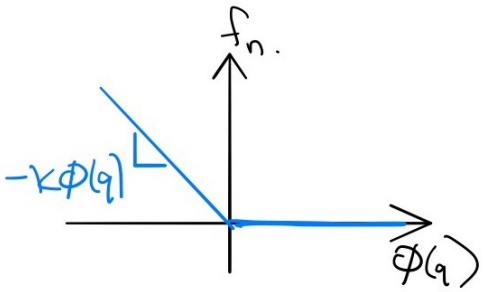
$$\mathbf{J}(\mathbf{q})=\begin{bmatrix}\mathbf{n}\\ \mathbf{t}_1\\ \mathbf{t}_2\end{bmatrix}。$$

按照写法，**Jv**给出了接触坐标中最接近的点的相对速度；它也可以用多三行来扩展，以输出完整的空间速度（例如在建立扭转摩擦模型时）。所产生的广义的力接触是由$J^T\lambda$给出的，其中$\lambda = [f_n, f_{t1}, f_{t2}]^T$。

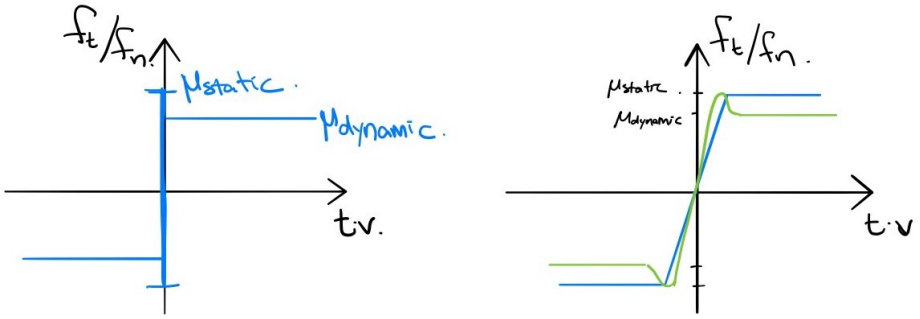
$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} + \mathbf{C}(\mathbf{q}, \mathbf{v})\mathbf{v} = \tau_g(\mathbf{q}) + \mathbf{B}\mathbf{u} + \mathbf{J}^T(\mathbf{q})\lambda。 \quad (20)$$

B.3.1 符合标准的接触模式

大多数顺应性接触模型在概念上是直截了当的：我们将使用一个硬Æ弹簧（和阻尼器[12]）来实现接触力，产生抵抗穿透的力（并对碰撞和/或摩擦的耗散进行粗略建模）。例如，对于法向力，f_n，我们可以使用一个简单的（片状）线性弹簧定律。



库仑摩擦是由两个参数描述的，即μ_{static}和$\mu_{dynamic}$，它们是静态和动态摩擦的系数。当接触切向速度（由**tv**给定）为零时，摩擦力将产生抵抗运动所需的任何力量，直到阈值$\mu_{static}f_n$。 但一旦超过这个阈值，我们就有滑移（接触切向速度不为零）；在滑移过程中，摩擦力将在反对运动的方向上产生一个恒定的拖力$|f_t|=\mu_{dynamic}f$。 这种行为不是当前状态的简单函数，但我们可以用一个连续的函数来近似它，如下图。

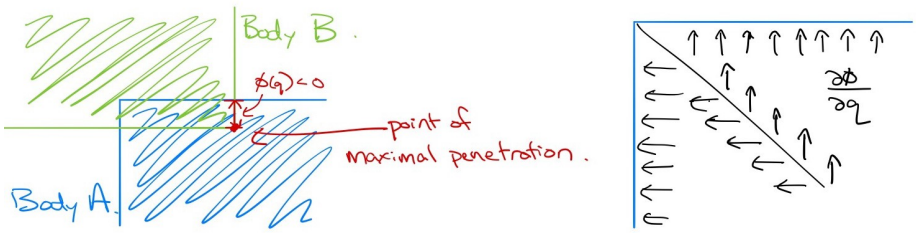


图B.5 -（左） Coloumb摩擦模型。(右)摩擦的连续片断-线性近似(绿色)和Coloumb摩擦的Stribeck近似(蓝色)；x轴是接触切向速度，y轴是摩擦系数。

有了这两个定律，我们可以把接触力恢复为当前状态的相对简单的函数。然而，魔鬼就在细节中。这种方法有几个特点，可以使它在数值上不稳定。如果你曾经在机器人模拟器中工作，看着你的机器人走了一步，却从地面上爆炸开来，在空中飞驰，你就知道我的意思了。

为了紧密地接近大多数机器人与世界的（几乎）刚性接触，接触 "弹簧 "的硬度必须相当高。例如，我可能希望我的180公斤的仿人机器人模型在稳态站立时插入地面的距离不超过1毫米。在我们的模型中加入stiÆ弹簧的挑战是，这将导致stiÆ微分方程（*stiU这个*词在弹簧和ODE中都是常用术语，这并不是巧合）。因此，顺应性接触模拟的最佳实现方式是使用特殊用途的数值积分配方（例如[13]），而且顺应性接触模型往往难以用于轨迹/反馈优化等方面。

但是这个模型的基本实现还有一个严重的数值挑战。计算带符号的距离函数$\phi(\mathbf{q})$，当它是非负值时是很简单的，但一旦两个物体发生碰撞，稳健地计算 "穿透深度 "就不简单了。考虑用$\phi(\mathbf{q})$来表示两个相互接触的盒子的最大穿透深度的情况。在顺应性接触中，我们必须有一些穿透力才能产生任何接触力。但是法向量的方向，$\mathbf{n} = \frac{\partial \phi}{\partial \mathbf{q}}$，很容易随着最大穿透点的移动而发生不连续的变化，正如我在这里试图说明的那样。



图B.6 -（左） 穿透力中的两个盒子，签名距离由最大穿透深度决定。(右)各种最大穿透点的接触法线。

如果你仔细想想这个例子，它实际上更多的是试图一致地确定接触点和法线的缺陷。将身体B上的点定义为对身体A的最大穿透点似乎是合理的，但注意到我所画的，这实际上并不是唯一的！身体A上的对应点可能是表面上距离最小的点。身体A上的对应点可能应该是表面上与最大穿透点距离最小的点（其他诱人的选择可能会导致距离在穿透开始时是不连续的）。但这整个策略是不对称的；为什么我不应该选择对身体B有最大穿透力的矢量？我的观点是，这些情况是存在的，而且即使我们最好的

当你研究细节时，软件的实现变得相当不令人满意。而在实践中，期望碰撞引擎在每一种情况下都能给出一致和平滑的接触点是很难的。

这种方法的一些优点包括：（1）它很容易实现，至少对于简单的几何形状而言；（2）由于是连续时间模型，它可以用误差控制的积分器进行模拟；（3）"刚性 "接触的近似的严格程度可以通过相对直观的参数来控制。然而，处理穿透力的数值挑战是非常现实的，它们促使我们采用另外两种方法，试图更严格地执行非穿透力约束。

B.3.2 带有事件检测的刚性接触 冲动碰撞

碰撞事件由以下的零交叉点（从正到负）来描述
ϕ.让我们首先假设无摩擦碰撞，让我们写出

$$\mathbf{M}\ddot{\mathbf{q}} = \tau + \lambda \mathbf{n}^T。 \tag{21}$$

其中$\mathbf{n}$是我们上面定义的 "接触法线"，$\lambda \geq 0$现在是一个（标量）冲动力，当在碰撞时间上进行整合时，冲动力是很好定义的
(表示为t^-到t^+）。将方程的两边整合到该（瞬时）上
间。

$$\int_{t_c^-}^{t_c^+} dt\,[\mathbf{M}\ddot{\mathbf{q}}] = \int_{t_c^-}^{t_c^+} dt\,[\tau + \lambda \mathbf{n}^T]$$

由于$\mathbf{M}$、τ和$\mathbf{n}$在这个区间内是常数，我们剩下的就是

$$\mathbf{M}\dot{\mathbf{q}}^{+} - \mathbf{M}\dot{\mathbf{q}}^{-} = \mathbf{n}^T \int_{t_c^-}^{t_c^+} \lambda dt。$$

其中$\dot{\mathbf{q}}^+$是$\dot{\mathbf{q}}(t^+)$的简写。将两边都乘以$\mathbf{n}\mathbf{M}^{-1}$，我们有

$$\dot{\mathbf{q}}^{+} - \dot{\mathbf{q}}^{-} = \mathbf{n}\mathbf{M}\mathbf{n}^{-1T} \int_{t_c^-}^{t_c^+} \lambda dt。$$

碰撞后，我们有$\dot{j}^+ = -\dot{e}^-$，其中$0 \leq e \leq 1$表示恢复原状的coeÆicient，得出。

$$\dot{\mathbf{q}}^{+} - \dot{\mathbf{q}}^{-} = -(1+e)\dot{\mathbf{q}}^{-}。$$

因此

$$\int_{t_c^-}^{t_c^+} \lambda dt = -(1+e)\left[\mathbf{n}\mathbf{M}\mathbf{n}^{-1T}\right]^{\#}\dot{\mathbf{q}}^{-}。$$

我在$^{\#}$这里用A来表示A的伪逆（通常我会写成A^+，但为了避免混淆，在本节中改变了它）。将其代入上文，结果为

$$\dot{\mathbf{q}}^{+} = \left(\mathbf{I} - (1+e)\mathbf{M}\mathbf{n}^{-1T}\left[\mathbf{n}\mathbf{M}\mathbf{n}^{-1T}\right]^{\#}\mathbf{n}\right)\dot{\mathbf{q}}^{-}。 \tag{22}$$

我们可以在接触处增加摩擦力。刚性体几乎总是只在一个点上经历接触，所以我们通常忽略扭转摩擦[...8]，而只对切向摩擦进行建模，将$\mathbf{n}$扩展为一个矩阵

$$\mathbf{J} = \begin{bmatrix} \mathbf{n} \\ \mathbf{t}_1 \\ \mathbf{t}_2 \end{bmatrix}.$$

来捕捉与接触面相切的接触位置的梯度。然后，λ成为代表接触冲力的笛卡尔矢量，（对于无摩擦）冲击后速度条件成为$\dot{\mathbf{q}}^+ = \text{diag}(-e, 0, 0)\dot{\mathbf{q}}^-$，导致方程式。

$$\dot{\mathbf{q}}^+ = \left(\mathbf{I} - \mathbf{M}\mathbf{J}^{-1T}\,\text{diag}(1+e, 1, 1)\left[\mathbf{J}\mathbf{M}\mathbf{J}^{-1T}\right]^{\#}\mathbf{J}\right)\dot{\mathbf{q}}^-. \tag{23}$$

如果λ被限制在一个摩擦锥上，就像在库仑摩擦中一样，一般来说，我们必须在不等式约束下求解优化来解决$\dot{\mathbf{q}}^+$的问题。

例B.3（一个在地面上弹跳的旋转球）。

想象一下，在平面上有一个质量为m、半径为r的球（一个空心球），其结构由$q = [x, z, \vartheta]$给出^T。运动方程为

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & \frac{2}{3}mr^2 \end{bmatrix}\ddot{\mathbf{q}} = \begin{bmatrix} 0 \\ -mg \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ r & 0 \end{bmatrix}\begin{bmatrix} \lambda_z \\ \lambda_x \end{bmatrix} = \tau_g + \mathbf{J}^T\lambda,$$

其中λ是接触力（除了在冲动碰撞过程中为零）。给定一个恢复力系数e和与水平地面的碰撞，碰撞后的速度为：。

$$\dot{\mathbf{q}}^+ = \begin{bmatrix} \frac{3}{5} & 0 & -\frac{2}{5}r \\ 0 & -e & 0 \\ \frac{3}{5r} & 0 & \frac{2}{5} \end{bmatrix}\dot{\mathbf{q}}^-.$$

把它放在一起

我们可以把双边约束方程和冲动碰撞方程放在一起，实现形式为单边约束的混合模型。让我们来概括一下

$$\phi(\mathbf{q}) \geq 0, \tag{24}$$

为刚体之间所有成对（有符号）距离函数的向量；这个向量描述了所有可能的接触。在每个时刻，一些接触的子集是活跃的，可以被视为双边约束（$j_i = 0$）。混合模型的守卫是当一个不活跃的约束变得活跃（$j_i > 0 \rightarrow j_i = 0$），或者当一个活跃的约束变得不活跃（$\lambda_i > 0 \rightarrow \lambda_i = 0$ 和 $\dot{j}_i > 0$）。请注意
碰撞事件将（几乎总是）只导致新的主动约束，当$e = 0$
例如，我们有纯非弹性碰撞，因为弹性碰撞很少允许持续接触。

如果接触涉及库仑摩擦，那么粘着和滑动之间的过渡可以被建模为额外的混合守卫。

B.3.3 刚性接触的时间步进近似值

到目前为止，我们已经看到两种不同的方法来执行接触的不等式约束，$j(\mathbf{q}) \geq 0$摩擦锥。首先我们介绍了顺应性接触，它有效地执行了非穿透性的硬弹簧。然后我们讨论了事件检测作为在不同的模型之间切换的手段，这些模型将主动约束视为平等约束。但是，也许令人惊讶的是，其中最重要的一点是

流行的和可扩展的方法是不同的：它涉及到制定一个数学程序，可以直接解决模拟的每个时间步骤的不等式约束。这些算法在每个时间步骤上的计算可能更昂贵，但它们允许在潜在的更大和更一致的时间步骤上进行稳定的模拟。

互补性的提法

我们需要哪一类数学程序来模拟接触？接触的离散性表明，我们可能需要某种形式的组合优化。事实上，最常见的转述是线性互补问题（LCP）[14]，由[15]和[16].一个LCP通常被写成

$$\begin{array}{ll} \text{发现} & \mathbf{w}=\mathbf{q}+\mathbf{M}\mathbf{z}。 \\ \text{纬度} & \mathbf{w} \geq \mathbf{z}0, \geq \mathbf{w}0,^{T}\mathbf{z}=0. \end{array}$$

它们与（潜在的非凸）二次方程序的最优性条件直接相关，并且[16]>表明，由我们的刚体接触问题产生的LCP可以用Lemke的算法来解决。作为简称，我们将这些互补性约束写为

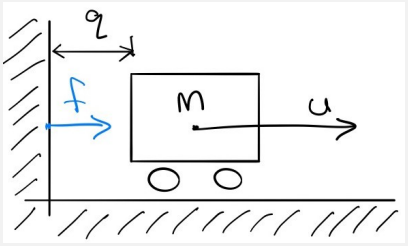
$$\begin{array}{ll} \text{发现} & \\ \mathbf{z} & \text{受制于} \leq 0(\mathbf{q}+\mathbf{M}\mathbf{z}) \perp \mathbf{z} \geq 0. \end{array}$$

让我从两个非常简单的例子开始，而不是深入研究完整的转录，其中有许多术语，而且可能相对地难以解析。

例B.4（时间步长的LCP：法向力）。

考虑我们最喜欢的简单质量被一个水平力驱动（用双积分器动力学），但这次我们将添加一堵墙，防止小车位置出现负值：我们的非穿透约束是$q \geq 0$。从物理学上讲，这个约束是由一个法向（水平）力f来实现的，从而产生运动方程。

$$mq^{\cdot\cdot} = u + f。$$



f被定义为执行非穿透性约束所需的力；当然，以下情况是真实的：$f \geq 0$和$q \cdot f = 0$。$q \cdot f = 0$是"互补性约束"，你可以把它理解为"要么q为零，要么力为零"（或者两者都是）；它是我们的"没有距离的力"约束，而且它显然是非凸的。事实证明，满足这些约束条件，再加上$q \geq 0$，是完全去掉f的足够条件。

为了确定LCP，我们首先将时间离散化，通过近似计算

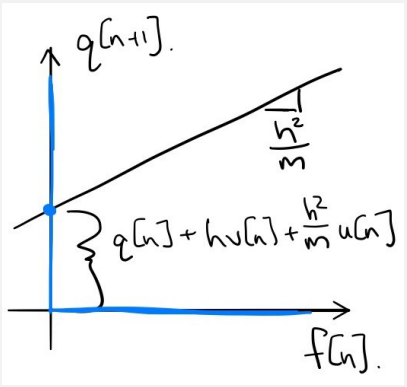
$$\begin{aligned} q[n+1] &= q[n] + hv[n+1]。 \\ v[n+1] &= v[n] + \frac{h}{m}(u[n]+f[n])。 \end{aligned}$$

这几乎是标准的欧拉近似，但请注意在第一个方程的右侧使用v[n+1]--这实际上是一个半隐式的欧拉近似，这个选择在推导中至关重要。

给出h、 $q[n]$、 $v[n]$和u[n]， 我们可以同时求解f[n]和q[n+1]， 方法是解决以下LCP。

$$q[n+1] = [q[n] + hv[n] + \frac{h^2}{m}u[n]] + \frac{h^2}{m}f[n]$$
$$q[n+1] \geq 0, \quad f[n] \geq 0, \quad q[n+1] \cdot f[n] = 0.$$

花点时间， 说服自己这是在上面给出的LCP处方中Øts。



注意：请不要将这种可视化与更常见的以 "互补锥 "为单位的LCP（二维或多维）的解空间的可视化相混淆[14].

也许'绘制解决方案、q[n+1]、$f[n]$作为q[n]、 $v[n]$的函数也有帮助。我在这里做了m = h1, = u0.1,[n] = 0:

在（时间步长，"脉冲速度"）LCP表述中，我们将接触问题写成它的组合形式。在上面的简单例子中，互补性约束迫使任何解决方案位于正x轴（$f[n]\geq 0$）*或*正y轴（$q[n+1]\geq 0$）上。 平等性约束只是一条将与该约束流形相交于解的线。但在这种无摩擦的情况下，重要的是要认识到这些只是一个的最优条件。

凸优化问题：我们的高斯原理的离散时间版本。

以上使用。用$\mathbf{v}'$作为$\mathbf{v}[n+1]$的缩写， 并将$\ddot{\mathbf{q}} = \frac{\mathbf{v}'-\mathbf{v}}{h}$ 在公式(13)和将目标按h缩放^2， 我们有。

$$\text{闵}_{\mathbf{v}'} \quad \frac{1}{2}(\mathbf{v}'-\mathbf{v}-\mathbf{h}\mathbf{M}^{-1}\tau)^T\mathbf{M}(\mathbf{v}'-\mathbf{v}-\mathbf{h}\mathbf{M}^{-1}\tau)$$
$$\text{受制于} \quad \frac{1}{h}j(\mathbf{q}') = \frac{1}{h}j(\mathbf{q}+\mathbf{h}\mathbf{v}') \approx \frac{1}{hhh}j(\mathbf{q})+\mathbf{n}\mathbf{v}' \geq 0.$$

如果我们将接触冲力应用之前的下一步速度表示为$\mathbf{v}^- = \mathbf{v} + \mathbf{h}\mathbf{M}^{-1}\tau$， 那么目标就更干净/更直观了。

$$\text{闵}_{\mathbf{v}'} \quad \frac{1}{2}(\mathbf{v}'-\mathbf{v}^-)^T\mathbf{M}(\mathbf{v}'-\mathbf{v}^-)$$
$$\text{受制于} \quad \frac{1}{h}j(\mathbf{q}') \geq 0.$$

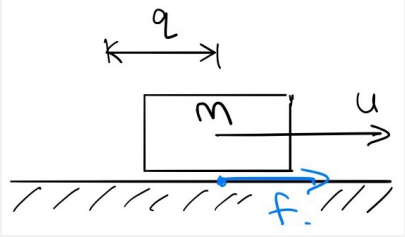
无摩擦接触动力学的LCP正是这个凸(因为$\mathbf{M}$=0)二次程序的优化条件， 而接触脉冲$\mathbf{f}\geq 0$再次扮演了拉格朗日乘数的角色(单位为$N\cdot s$)。

那么， 为什么我们这么多地谈论LCP而不是QP？LCP也可以代表更广泛的问题类别，这就是我们用标准转录法得出的结果

的库伦摩擦。在有限的摩擦中，我们可以增加一个额外的约束，即每个接触点的切向速度等于零（但这些方程不一定有可行的解决方案）。一旦我们承认对摩擦力大小的限制，二元形式的非凸性就会抬头。

例B.5（时间步长的LCP：库仑摩擦）。

我们可以用LCP来找到一个具有库仑摩擦力的可行方案，但这需要对松弛变量做一些体操来使其发挥作用。特别是对于这种情况，我相信一个非常简单的例子是最好的。让我们拿起我们的砖头，去掉墙和轮子（所以我们现在与地面有摩擦）。



其动态变化与我们之前的例子相同。

$$mq^{\cdot\cdot} = u + f。$$

但这次我用f表示摩擦力，如果$q^{\cdot}=0$，摩擦力在摩擦锥内，如果$q^{\cdot}\neq 0$，摩擦力在摩擦锥的边界上；这被称为最大耗散原理[15].这里，法向力的大小总是mg
所以我们有$|f|\leq \mu mg$，其中μ是摩擦系数。 我们将使用同样的半隐式欧拉近似法将其转化为离散时间。

现在，为了将摩擦约束写成LCP，我们必须引入一些松弛变量。首先，我们将把力分成一个正分量和一个负分量：$f[n] = f^{+} - f^{-}$。我们还将引入一个变量v_{abs}，如果速度不为零（在任何一个方向），它将是不为零的。现在我们可以写出LCP。

$$\begin{array}{ll} & \text{发现} \quad \text{受制于} \\ {}_{v_{abs}} f^{+} f^{-} & \\ 0 \leq v_{abs} + v[n+1] \perp f^{+} \geq 0 & 0 \leq v_{abs} - v[n+1] \perp f^{-} \geq 0 \\ \mu 0 & 0 mg - f^{+} - f^{-} \perp v_{abs} \geq 0, \end{array}$$

其中$v[n+1]$的每个实例我们都用以下方式写出来

$$v[n+1] = v[n] + \frac{h}{m}(u + f^{+} - f^{-})。$$

这足以让你头晕目眩!但如果你把它算出来，我们想要的所有约束都在那里。 例如，对于$v[n+1]>0$，那么我们必须有$f = +0$，$v_{abs} = v[n+1]$，并且$f = -\mu mg$。当$v[n+1]=0$，我们可以有$v_{abs} = 0, f^{+} - f^{-} \leq \mu mg$，而这些力量加起来必须使$v[n+1]= 0$。

我们可以把这一切放在一起，写出一个具有法向力和摩擦力的LCP，通过库仑摩擦力（使用三维摩擦锥的多面体近似）进行关联[15].尽管任何LCP的解也可以表示为QP的解，但这个问题的QP是不凸的。

阿尼特斯库的凸式公式

在[17]中，Anitescu通过放弃最大耗散不等式和允许摩擦锥内的任何力，描述了这个问题的自然凸表述。考虑以下的优化。

$$\begin{aligned} &\min_{\mathbf{v}'} \quad \frac{1}{2}(\mathbf{v}'-\mathbf{v}^-)^T\mathbf{M}(\mathbf{v}'-\mathbf{v}^-) \\ &\text{受制于} \quad \frac{1}{h}j(\mathbf{q}') + \mu\mathbf{d}_i\mathbf{v}' \geq 0, \qquad \forall i \in 1,\ldots,m \end{aligned}$$

其中$\mathbf{d}_i$，$\forall i \in 1, \ldots, m$是一组联合坐标中的切线行向量，对摩擦锥的多面体近似进行参数化（如[15]）.重要的是，对于每一个$\mathbf{d}_i$，我们也必须有$-\mathbf{d}_i$在这个集合中。通过书写拉格朗日。

$$L(\mathbf{v}', \beta) = \frac{1}{2}(\mathbf{v}'-\mathbf{v}^-)^T\mathbf{M}(\mathbf{v}'-\mathbf{v}^-) - \sum_i \beta_i\left(\frac{\varphi_i(\mathbf{q})}{h} + (\mathbf{n}+\mu\mathbf{d})\mathbf{v}'\right) 。$$

并检查静止性条件。

$$\frac{\partial L}{\partial \mathbf{v}'}^T = \mathbf{M}(\mathbf{v}'-\mathbf{v}) - h\tau - \sum_i \beta_i(\mathbf{n}+\mu_i\mathbf{d})^T \quad 0,$$

我们可以看到，与第1个约束相关的拉格朗日乘数$\beta_i \geq 0$是$\mathbf{n}-\mu\mathbf{d}$方向i上的冲力（单位为$N\cdot s$）的大小，其中$\mathbf{n}=$是$\frac{\partial\varphi}{\partial\mathbf{q}}$接触法线；力的总和是多面体摩擦锥的这些极端射线的阿埃尼组合。按照写法，优化是一个QP。

但要注意！这是不可能的。虽然原始解是凸的，而且对偶问题总是凸的，但这里的目标可以是正的半非线性的。这并不是一个错误
-- [17] 描述了几个简单的例子，其中$\mathbf{v}'$的解是唯一的，但
而产生它的冲动则不是（想想一张有四条腿的桌子）。

当切向速度为零时，这个约束是严格的；对于滑动接触来说，松弛有效地导致接触 "水压 "脱离接触，因为$j(\mathbf{q}') \geq h\mu_i\mathbf{d}\mathbf{v}'$。 对我来说，这似乎是一个相当合理的近似值，特别是对于
小H!

让我们继续下去，写出对偶程序。为了使符号更清晰，我们将雅各布向量堆叠到$\mathbf{J}$中β，这样第$\mathbf{i}$行是$\mathbf{n}+\mu\mathbf{d}_\mathbf{i}$，所以我们有

$$\frac{\partial L'}{\partial \mathbf{v}'}^T = \mathbf{M}(\mathbf{v}-\mathbf{v}) - h\tau - \mathbf{J}_\beta^T\beta = 0.$$

将其代入拉格朗日，就可以得到对偶程序。

$$\min_{\beta \geq 2} \frac{1}{2}\beta^T\mathbf{J}_\beta\mathbf{M}\mathbf{J}_\beta^{-1T}\beta + \frac{1}{h}\phi_0(\mathbf{q})\sum_i \beta_i 。$$

最后一点：我在这里只写了一个接触点，以简化符号，但当然我们对每个接触点都要重复约束条件；[17]研究的是只包括$j(\mathbf{q}) \leq \varepsilon$的潜在接触对的典型情况，对于小$\varepsilon \geq 0$。

托多罗夫的凸式公式

根据[18]，流行的MuJoCo模拟器使用了这种凸松弛的轻微变化，进行了优化。

$$\begin{aligned} &\min_{\lambda} \quad \frac{1}{2}\lambda^T\mathbf{J}\mathbf{M}\mathbf{J}^{-1T}\lambda + \frac{1}{h}\lambda^T(\mathbf{J}\mathbf{v}^- - \dot{\mathbf{x}}^d) 。 \\ &\text{受制于}\lambda \in FC(\mathbf{q})。 \end{aligned}$$

其中$\dot{\mathbf{x}}^d$是一个 "期望的接触速度"，$FC(\mathbf{q})$描述了摩擦锥。 摩擦锥约束看起来很新，其实不然；观察$J^T\lambda = J^T\beta$，其中$\beta \geq 0$

是多面体情况下摩擦锥的一个巧妙的参数化。更大的差异在于线性项：[18] 提出$\dot{\mathbf{x}}^d = \mathbf{Jv} - \mathbf{hBJv} - \mathrm{hK}[j(\mathbf{q}), 0, 0]^T$， B和K的稳定增益是为了产生临界阻尼响应。

我们应该如何解释这个？如果你展开线性条款，你会发现这几乎正是我们从位置平等约束的表述中得到的对偶形式，包括鲍姆加特稳定。思考一下这种表述的含义是很有意思的--就像阿尼特斯库松弛一样，有可能得到一些 "距离上的力"，因为我们没有以任何方式表达λ=当$0 j(\mathbf{q})>0$. 在阿尼特斯库，它发生在一个与速度有关的边界层中；在这里，如果你 "热身"，它可能发生--以足够的相对速度走向接触，稳定化条款想要使你减速。

在对偶形式中，自然要考虑摩擦锥的全锥体描述。

$$\mathrm{FC}(\mathbf{q}) = \{\boldsymbol{\lambda} = [f_n, f_{t1}, f_{t2}]^T \; f_n \geq 0, \sqrt{f_1^2 + f_2^2} \leq \mu_n f\}。$$

由此产生的双重优化有一个二次元目标和二阶锥体约束（SOCP）。

在MuJoCo中还有其他一些重要的放松措施。为了解决目标中的正不确定性，[18] 通过在对角线上增加一个小项来放松对偶目标。这保证了凸性，但凸性优化的条件仍然很差。目标中的稳定项是另一种形式的放松，而[18]还实现了对惯性矩阵增加额外的稳定化，称为 "隐性阻尼惯性"。这些松弛可以极大地提高仿真速度，既可以使每个凸优化的求解速度更快，又可以使仿真器采取更大的积分时间步长。MuJoCo拥有一个特殊用途的求解器，可以以非常惊人的速度模拟复杂的场景--通常比实时速度快几个数量级。但这些放松也可能被滥用--不幸的是，经常可以看到物理上荒谬的MuJoCo模拟，特别是当那些不太关心物理的研究人员开始改变模拟参数以优化他们的学习曲线时。

超越点接触

即将推出...飞机上的盒子的例子。多接触。

另外，单点力不能捕捉到像扭转摩擦这样的效应，在一些非常简单的情况下（像一个坐在桌子上的盒子）表现得很差。因此，许多最有效的接触算法将自己限制在非常有限/简单的几何形状上。例如，我们可以把 "点接触"（零半径球体）放在机器人脚的四个角上；在这种情况下，当它们穿透其他物体/网格时，在这四个点上增加力，可以得到更一致的接触力位置/方向。很多模拟器，特别是腿部机器人的模拟器，都是这样做的，令人惊讶。

在实践中，大多数碰撞检测算法会在给定的身体列表中返回一个 "碰撞点对 "的列表（每对碰撞的身体有一个点对，如果使用前面提到的 "多接触 "启发式算法，则有更多的点对），而我们的接触力模型只是在这些点对之间附加弹簧。给出一个点对，p_A和p_B，都在世界坐标中。...

德雷克的水弹性模型[19].

B.4 静态行动原则

[20] 说

> 最小行动的原则--实际上是静止行动的原则--是

是经典物理学定律的最紧凑形式。这个简单的规则（可以用一行字来写）总结了一切！不仅是经典力学的原理，还有电磁学、广义相对论、量子力学、化学的一切知识--直到最终的化学反应。不仅是经典力学的原理，还有电磁学、广义相对论、量子力学、关于化学的一切已知的东西--直到物质的最终已知成分，基本粒子。

听起来很重要!

我发现很多关于拉格朗日的静止作用原理推导的介绍都是不必要的混乱。下面是我的版本，希望能有所帮助。

考虑一个在$t\in[t_0,\ t]$上$_1$定义的轨迹$\mathbf{q}(t)$。静态作用原理指出，如果该轨迹代表了 "作用 "的一个静止点，那么它就是治理我们系统的二阶梯方程的有效解，该方程被定义为

$$A = \int_{t_0}^{t_1} L(\mathbf{q},\ \dot{\mathbf{q}})\, dt。$$

一个轨迹是一个静止点是什么意思？使用变化微积分，我们认为这个轨迹的一个极小的变化：$\mathbf{q}(t)+\varepsilon(t)$，其中$\varepsilon(t)$是一个任意的二可函数，在$t_0$和$t$处为零$_1$。这个变化不应该改变行动，这意味着$\delta A$=对于0任何小的$\varepsilon(t)$。

$$\delta A = A(\mathbf{q}(t) + \varepsilon(t) - A(\mathbf{q}(t))$$
$$= \int_{t_0}^{t_1} L(\mathbf{q}(t) + \varepsilon(t),\ \dot{\mathbf{q}}(t) + \dot{\varepsilon}(t)) dt - \int_{t_0}^{t_1} L(\mathbf{q}(t),\ \dot{\mathbf{q}}(t))\, dt。$$

我们可以扩展第一个积分中的项。

$$L(\mathbf{q}(t) + \varepsilon(t),\ \dot{\mathbf{q}}(t) + \dot{\varepsilon}(t)) = L(\mathbf{q}(t) + \varepsilon(t), \dot{\mathbf{q}}(t)) + \frac{\partial L}{\partial \mathbf{q}(t)}\varepsilon(t) + \frac{\partial L}{\partial \dot{\mathbf{q}}(t)}\dot{\varepsilon}(t)。$$

将此代入，并将叶子简化。

$$\delta A = \int_{t_0}^{t_1} \left[\frac{\partial L}{\partial \mathbf{q}(t)}\varepsilon(t) + \frac{\partial L}{\partial \dot{\mathbf{q}}(t)}\dot{\varepsilon}(t)\right] dt。$$

现在对第二项使用分项积分法。

$$\int_{t_0}^{t_1} \frac{\partial L}{\partial \dot{\mathbf{q}}(t)}\dot{\varepsilon}(t)dt = \left[\frac{\partial L}{\partial \dot{\mathbf{q}}(t)}\varepsilon(t)\right]_{t_0}^{t_1} - \int_{t_0}^{t_1} \frac{d}{dt}\frac{\partial L}{\partial \dot{\mathbf{q}}(t)}\varepsilon(t)dt。$$

并观察到右手边的第一项是零，因为$\varepsilon(t_0)=\varepsilon(t_1)= 0$。这使得

$$\delta A = \int_{t_0}^{t_1} \left[\frac{\partial L}{\partial \mathbf{q}} - \frac{d}{dt}\frac{\partial L}{\partial \dot{\mathbf{q}}}\right]\varepsilon(t)dt = 0。$$

由于这对任何$\varepsilon(t)$都必须积分为零，我们必须有

$$\frac{\partial L}{\partial \mathbf{q}} - \frac{d}{dt}\frac{\partial L}{\partial \dot{\mathbf{q}}} = 0,$$

可以乘以负1，得到熟悉的（非强制）拉格朗日运动方程的形式。强制版本是由变化而来的

$$\delta A = \int_{t_0}^{t_1} L(\mathbf{q}(t),\ \dot{\mathbf{q}}(t)) dt + \int_{t_0}^{t_1} \mathbf{Q}^T(t)\varepsilon(t)dt = 0.$$

关于更丰富的处理方法，见[2].

参考文献

1. John Craig, "Introduction to Robotics:机械学和控制",Addison Wesley,一月 。 1989.
2. Cornelius Lanczos, "The variational principles of mechanics" ,University of Toronto Press , no. 1970.
3. H.浅田和J.E. Slotine, "机器人分析与控制", , 第93-131页。 1986.
4. Roy Featherstone, "EÆicient Factorization of the Joint Space Inertia Matrix for Branched Kinematic Trees", *International Journal of Robotics Research*, vol. no24,. pp6,.487-500, 2005.
5. H.Choset and K. M. Lynch and S. Hutchinson and G. Kantor and W. Burgard and L. E. Kavraki and S. Thrun, "Principles of Robot Motion-Theory, Algorithms, and Implementations", The MIT Press , 2005.
6. Jean-Jacques E. Slotine and Weiping Li, "Applied Nonlinear Control", Prentice Hall, October, 1990.
7. Vincent Duindam, "Port-Based Modeling and Control for EÆicient Bipedal Walking Robots", 博士论文, 特文特大学, 3月。 2006.
8. Roy Featherstone, "刚体动力学算法",Springer, 2007.
9. Brian Mirtich, "基于脉冲的刚体系统动态模拟", 博士论文, 加州大学伯克利分校。 1996.
10. Abhinandan Jain, "Robot and Multibody Dynamics:分析与算法",Springer US, 2011.
11. Firdaus E Udwadia和Robert E Kalaba, "A new perspective on constrained motion", *Proceedings of the Royal Society of London. 系列A：数学和物理科学* , 第439,407-410页1906,。 1992.
12. K.H. Hunt和F. R. E. Crossley, "CoeÆicient of restitution interpreted as damping in vibroimpact", *Journal of Applied Mechanics*, vol. 42 Series E, no. 440-445页2,。 1975.
13. Alejandro M Castro and Ante Qu and Naveen Kuppuswamy and Alex Alspach and Michael Sherman, "A Transition-Aware Method for the Simulation of Compliant Contact with Regularized Friction", *IEEE Robotics and Automation Letters*, vol. no5,. pp2,.1859-1866。 2020.
14. Katta G Murty和Feng-Tien Yu, "线性互补性、线性和非线性编程",Citeseer,Vol. 3,1988.
15. D.E. Stewart和J.C. Trinkle, "AN IMPLICIT TIME-STEPPING SCHEME FOR RIGID BODY DYNAMICS WITH INELASTIC COLLISIONS AND COULOMB FRICTION", *International Journal for Numerical Methods in Engineering*, vol. no39,. pp15,.2673-2691。 1996.
16. M.Anitescu和F.A. Potra, "将有摩擦的动态多刚体接触问题表述为可解决的线性互补问题", 《*非线性动力学*》, 第14,231-247页3,。 1997.
17. Mihai Anitescu, "基于优化的非光滑刚性多体动力学模拟", 《*数学编程*》, 第113--143页1,, Jan105,, 2006.
18. Emanuel Todorov, "具有接触和约束的凸和可分析的动力学。在{MuJoCo}

中的理论和实现"。 *2014*

IEEE国际机器人和自动化会议（ICRA），第6054-6061页。 2014.

19. Ryan Elandt and Evan Drumwright and Michael Sherman and Andy Ruina, "A pressure Øeld model for fast, robust approximation of net contact force and moment between nominally rigid objects", , pp.8238-824511,, 2019.

20. 伦纳德-苏斯金德和乔治-赫拉博夫斯基，"理论上的最低限度：你需要知道什么才能开始做物理学"，基本书籍。 2014.

上一章 目录 下一章 可访问性

低动能机器人技术

走路、跑步、游泳、 飞行和操纵的算法

吕斯-特德雷克

最后修改 2021-6-13.
如何引用这些笔记，使用注释，并给予反馈。

注意：这些是用于在麻省理工学院教授的课程的工作笔记。 它们将在2021年春季学期中被更新。 讲座视频可在YouTube上找到。

上一章 目录 下一章

附录C

在Colab中打开

优化和数学编程

这些笔记中对动力学和控制的看法在很大程度上建立在优化的工具上--我们在实践中的成功在很大程度上取决于对数值优化的有效应用。有许多关于优化的优秀书籍，例如[1]是一本关于平滑优化的优秀参考书。
[2]是一份关于凸优化的优秀参考资料（我们广泛使用）。 我将在下面提供更多关于具体优化公式的参考资料。

因此，本章的意图主要是提供一个启动平台，解决一些与机器人技术特别相关但在现有的一般处理中可能比较隐蔽的主题。作为这些工具的消费者，你只需有一个高层次的理解，就能很快走得很远。但是，就像许多事情一样，有时细节很重要，理解引擎盖下发生的事情变得很重要。我们经常可以用多种方式来表述一个优化问题，这些方式在数学上可能是等价的，但在实践中的表现却大相径庭。

C.1 优化软件

一些来自优化的算法自己实现起来相当简单；随机梯度下降也许是一个典型的例子。甚至更多的算法在概念上是相当简单的，但是对于一些算法来说，实施细节是非常重要的--就速度和/或鲁棒性而言，专家实施和新手实施的数值配方之间的差别可能是巨大的。这些软件包通常使用大量的技术来对问题进行数值调节，丢弃琐碎有效的约束条件，并在求解之间进行热启动优化。并非所有的求解器都支持所有类型的目标和约束，即使我们有两个商业求解器，它们都声称可以解决，例如二次方程序，那么它们在你的问题的特定结构/条件下可能会有不同的表现。在某些情况下，我们也有设计

得很好的商业求解器的开源替代品（通常是由优化方面的专家学者编写的）--有时它们与商业求解器竞争得很好，甚至在特定的问题类型中胜过它们。

因此，也有一些软件包试图在数学程序的制定和该问题在每个特定求解器中的实例化之间提供一个抽象层。这方面的著名例子包括MATLAB的CVX和YALMIP，以及Julia的JuMP。DRAKE的MathematicalProgram类为C++和Python提供了这样一个中间层；它的创建最初是由于需要支持我们在本文中使用的优化公式而特别激发的。

我们有很多关于DRAKE数学编程的教程，从这里开始有一个总体介绍。Drake支持一些定制的、开源的和商业的求解器（甚至一些商业求解器对学术用户是免费的）。

C.2 一般概念

一般的表述是 ...

重要的是要认识到，尽管这种表述方式具有难以置信的通用性，但它确实有其局限性。仅举一例，当编写优化程序来规划机器人的轨迹时，在这种表述中，我们通常必须先选择特定数量的决策变量来编码解决方案。尽管我们当然可以写出改变变量数量的算法，并再次调用优化器，但不知何故，我觉得这种 "通用 "表述未能捕捉到，例如，在基于样本的优化规划中出现的数学编程类型--在这种情况下，所产生的解决方案可以由任何Ønite数量的参数描述，并且计算在它们之间进行了∞exibly转换。

C.2.1 凸点与非凸点优化

局部优化。凸函数, 凸约束.

如果一个优化问题是非凸的，并不一定意味着优化是困难的。在深度学习中，有许多案例，我们可以可靠地解决看似非常高维和非凸的优化问题。我们对这些现象的理解正在迅速发展，我怀疑我在这里写的东西很快就会过时。但目前在监督学习中的思考大多围绕着这样的观点：过度参数化是关键--许多这样的成功案例都发生在我们实际上有比数据更多的决策变量的情况下，搜索空间实际上密布着能够完美Øt数据的解决方案（所谓的 "插值解决方案"），并非所有的全局最小值都是同样稳健的，优化算法正在执行某种形式的隐含正则化，以便挑选一个 "好的 "插值解决方案。

C.2.2 带拉格朗日乘数的限制性优化

给出平等约束的优化问题

$$\text{最小化}_{\mathbf{z}}\ell(\mathbf{z}),\ \text{条件是}\mathrm{j}(\mathrm{z})=0,$$

其中j是一个向量。确定一个拉格朗日乘数的向量λ，*其*大小与j相同，并确定标量拉格朗日函数。

$$L(\mathbf{z},\lambda)=\ell(\mathbf{z})+\lambda^{T}\phi(\mathbf{z})。$$

$\mathbf{z}^*$是受限优化的最优值的一个必要条件是L的梯度相对于$\mathbf{z}$和λ都消失了。

$$\frac{\partial L}{\partial \mathbf{z}}=0,\quad \frac{\partial L}{\partial \lambda}=0.$$

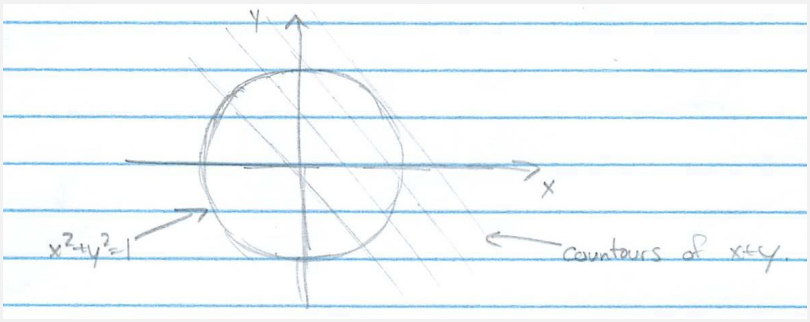
请注意，$\frac{\partial L}{\partial \lambda} = \phi(\mathbf{z})$，所以要求它为零就等于要求满足约束。

例C.1（单位圆上的优化）。

考虑以下优化。

$$\min_{x,y} x + y, \quad \text{subject to } x^2 + y^2 = 1.$$

$x+y$的水平集是斜率为−的1直线，而约束条件要求解住在单位圆上。



仅仅通过检查，我们可以确定，最佳解决方案应该是 $x = y = -\frac{\sqrt{2}}{2}$。让我们确认一下，我们可以用拉格朗日得到同样的结果乘法器。

制订

$$L = x + y + \lambda\,(x^2 + y^2 - 1)。$$

我们可以取导数并求解

$$\frac{\partial L}{\partial x} = +1\,\lambda 2x = 0 \quad \Rightarrow \quad \lambda = -\frac{1}{2x}$$

$$\frac{\partial L}{\partial y} = +\lambda 12y = 1 - \frac{y}{x} = 0 \quad \Rightarrow \quad y=x。$$

$$\frac{\partial L}{\partial \lambda} = x^2 + y^2 - 1 = 2x^2 - 1 = 0 \quad \Rightarrow \quad x = \pm\frac{1}{\sqrt{2}}$$

考虑到两个满足必要条件的解决方案，负解显然是目标的最小化。

C.3 凸面优化

C.3.1 LinearPrograms/QuadraticPrograms/SecondorderCones

例子。阿特拉斯的平衡力控制

C.3.2 半 独立编

程和线性矩阵不等式

例C.2（非凸二次约束的半极限编程放松）。

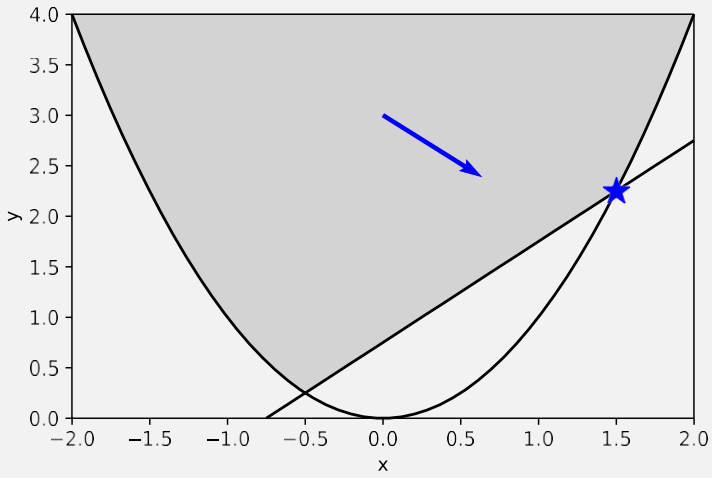
考虑一下这个问题。

$$\min_{x} /LX - A/L^2$$
$$\text{受制于} /lx - b/l \geq 1$$

我们可以把它写成

$$\min_{x,y} y - 2\,ax + a^2$$
$$\text{受制于} y - 2\,bx + b^2 \geq 1$$
$$y \geq x^2$$

在这里，我们把$y \geq x^2$写成半截式约束$\begin{bmatrix} y & x \\ x & 1 \end{bmatrix}$.



图C.2--a=b.8,=的优化景观 .5.

我用箭头标出了可行区域和目标。正如你所看到的，可行区域是与线性约束相交的y=x的图示²。现在是关键的一点：对于线性目标，只有当成本与目标直接正交时，最优解才会位于边界上；否则它将位于一个顶点上。因此，在这种情况下，只有在a=b的情况下，解决方案才会位于内部；对于其他任何数值，这种松弛都会给出最优解决方案。请注意，我们同样可以写一个二次平等的约束条件。

我认为这非常聪明，唯一令人沮丧的是它对二次函数有些特殊。(如果我们试图用同样的方法来处理例如在多面体外部的问题，那么二次方区域的整个外部可能是一个最优解，这一事实并不成立）。

C.3.3 二次方之和优化

事实证明，就像我们可以用SDP来搜索正二次方程一样，我们可以将其推广到搜索正多项式方程。为了清楚起见，对于二次方程，我们有

$$\mathbf{P} = 0 \quad \Rightarrow \quad \mathbf{x}^T\mathbf{P}\mathbf{x} \geq 0, \quad \forall \mathbf{x}.$$

事实证明，我们可以将其概括为

$$\mathbf{P} = 0 \quad \Rightarrow \quad \mathbf{m}^T(\mathbf{x})\mathbf{P}\mathbf{m}(\mathbf{x}) \geq 0。 \quad \forall \mathbf{x},$$

其中$\mathbf{m}(\mathbf{x})$是一个多项式方程的向量，通常选择*单项式*的向量（只有一个项的多项式）。以这种方式进行参数化的正多项式集合正是可以写成平方之*和的*多项式集合[3].虽然我们知道并非所有的正多项式都可以写成这种形式，但对于这个差距，我们知道得很多。对于我们的目的来说，这个差距是非常小的（已经有人写过关于试图找到均匀为正但不是平方之和的多项式的论文）；我们应该记住它的存在，但现在不要太担心它。

甚至更好的是，关于如何选择$\mathbf{m}(\mathbf{x})$中的项，有相当多的已知信息.例如，如果你给我一个多项式

$$p(x) = 2 - 4x + 5x^2$$

并问它是否对所有实数x*都*是正数，我可以通过产生平方和因式分解来说服你它是正数。

$$p(x) = +1(1-2x)^2 + x^2。$$

而且我知道我可以在我的单项式向量中不需要任何度数大于1的单项式（p的度数的平方根）来制定这个问题。在实践中，这对我们来说意味着，人们已经编写了优化前端，它接受了一个高层次的规格，对多项式的正性进行约束，并自动为你生成SDP问题，而无需考虑$\mathbf{m}(\mathbf{x})$中应该出现哪些项（参见[4]).这一类的优化问题被称为平方之和（SOS）优化。

碰巧的是，对$\mathbf{m}(\mathbf{x})$的特定选择会对所产生的半截式程序的数值产生巨大的影响（以及对你用商业求解器解决它的能力产生影响）。DRAKE实现了一些特别新颖/先进的算法，以使其工作良好[5].

我们将使用平方之和约束的优化方法写为

$$p(\mathbf{x}) \text{是SOS}$$

作为对所有$\mathbf{x}$的$p(\mathbf{x}) \geq 0$的约束的简写，正如通过找到一个平方之和分解所证明的。

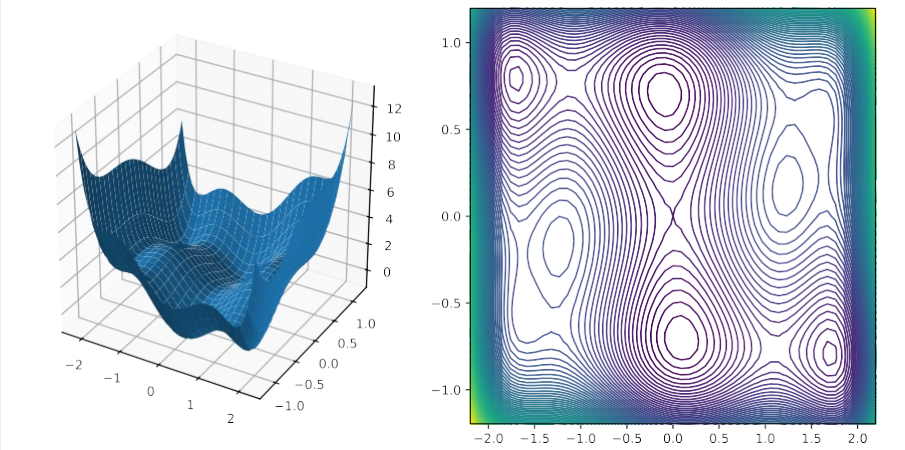
这令人惊讶地强大。它允许我们使用凸式优化来解决看似非常不凸的优化问题。

例C.3（通过SOS实现全球最小化）。

考虑以下著名的双变量非线性函数（称为 "六驼铃"。

$$p(x) = 4x^2 + xy - 4y^2 - 2.1x^4 + 4y^4 + x^6/3。$$

这个函数有六个局部最小值，其中两个是全局最小值[6].



通过制定一个简单的平方和优化，我们实际上可以通过写出这个函数的最小值（技术上，它只是一个下限，但在这种情况下和许多情况下，它是令人惊讶的严格）。

$$\max_{\lambda} \lambda$$
$$\text{s.t. } p(x) - \lambda \text{是sos。}$$

来吧，玩玩代码（大部分线条只是为了绘图；实际的优化问题是很好的、简单的制定）。

请注意，这Ønds最小值，但实际上并没有产生模仿它的**x**值。这是有可能的[6]，但这需要检查平方和解的对偶（我们在本章的目标中不需要）。

半代数集上的平方之和

S-程序。

代数变体上的平方和优化

S-程序

使用商环 通过采样的商环

卫星定位系统和特别卫星定位系统

C.3.4 解决方案技术

内点（Gurobi, Mosek, Sedumi, ...），一阶方法

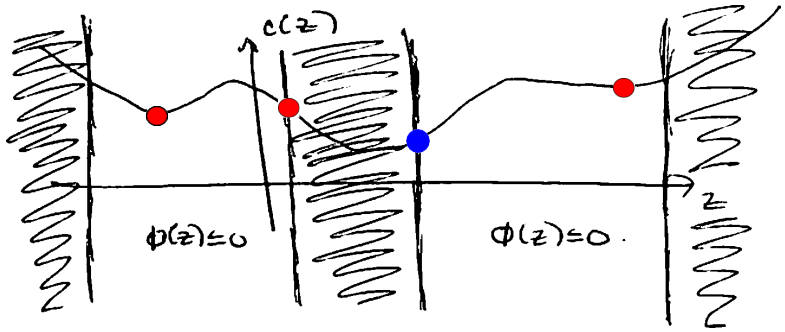
C.4 非线性编程

非线性优化问题的一般表述为

$$\min_{z} c(z) \text{ 在 } j(z) \le 0 \text{ 的条件下，}$$

其中，z是*决策变量*的一个向量，c是一个标量*目标函数*，j是一个*约束*向量。请注意，虽然我们写的是$j \le 0$，但这种表述方式包含了对决策变量的正向约束（只需将约束条件乘以-1）。
)和平等约束(简单地列出$j \le$和$0 - j \le 0$)，也是如此。

你脑子里的画面应该是一个非线性的、潜在的非凸的目标函数在（多维）z上的定义，有一个满足约束条件的可能的z值的子集。



图C.3 - 一个非线性优化问题的一维卡通。红点代表局部最小值。蓝点代表最优解。

请注意，最小值可能是目标函数具有零导数的结果。
*或*由于一个倾斜的目标对一个约束条件的影响。

解决这些优化问题的数字方法需要一个初始猜测，即z，并通过尝试将目标函数下移到最小值来进行。常见的方法包括梯度*下降法*，即计算或估计目标函数的梯度，或二阶方法，如*顺序二次编程（SQP）*，它试图对目标函数进行局部二次逼近，对约束条件进行局部线性逼近，并在每次迭代中解决二次程序，直接跳到局部逼近的最小值。

虽然不是严格要求，但这些算法通常可以从明确计算目标和约束条件的梯度中获益良多；另一种方法是通过数值纠偏获得梯度。除了纯粹的速度考虑，我更倾向于明确地计算梯度，因为它可以帮助避免数值精度问题，这些问题可能会在无偏移方法中悄悄出现。计算这些梯度的愿望将是下面讨论的一个主要主题，我们已经花了很大力气为我们提供的函数提供显式梯度，并为DRAKE中的用户提供的函数提供自动二分法。

当我开始工作时，我认为实现梯度下降甚至二阶方法没有任何diÆcult，而且我自己写了所有的求解器。现在我意识到我错了。可用于非线性规划的商业求解器比我自己写的任何东西的性能都要高得多，有许多技巧、微妙之处，以及参数选择，在实践中可以产生巨大的差异。其中一些求解器可以利用问题中的稀疏性（例如，如果约束条件以稀疏的方式作用于决策变量）。现在，我们最常使用的是SNOPT [...7]，它现在与DRAKE的二元分布捆绑在一起，但也支持一大套数值求解器。请注意，虽然我确实提倡使用这些工具，但你不需要把它们当作黑盒子来使用。在许多情况下，你可以通过了解和选择非默认的配置参数来提高优化性能。

C.4.1 二阶方法（SQP/内点法）

C.4.2 一阶方法（SGD/ADMM） 惩罚性方法

增强的拉格朗日

投影梯度下降

C.4.3 零阶方法（CMA）

C.4.4 例子。逆向运动学

C.5 组合优化

C.5.1 搜索，SAT，一阶逻辑，SMT求解器，LP解释

C.5.2 混合整数凸形优化

一本关于MIP的高级但非常可读的书[8].关于MILP的不错的调查报告[9].

C.6 "黑箱 "优化

无派生方法。有些允许有噪音的评价。

参考文献

1. Jorge Nocedal 和 Stephen J. Wright, "Numerical optimization",Springer, 2006.
2. StephenBoyd 和 LievenVandenberghe，《凸优化》，剑桥大学出版社。 2004.
3. Pablo A. Parrilo, "Robustness和Optimization中的Structured SemideØnite Programs and Semialgebraic Geometry Methods", 博士论文, 加州理工学院, 五月18, 2000.
4. Stephen Prajna和Antonis Papachristodoulou和Peter Seiler和Pablo A.Parrilo, "SOSTOOLS:用于MATLAB的平方之和优化工具箱用户指南", 61, 月 2004.
5. Frank Noble Permenter, "Reduction methods in semideØnite and conic optimization", PhD论文, Massachusetts Institute of Technology, 2017.[链接]
6. Didier Henrion and Jean-Bernard Lasserre and Johan Lofberg, "GloptiPoly 3: moments, optimization and semideØnite programming", *Optimization Methods \& Software*, Vol. no24,.4-5, pp.761-779, 2009.
7. Philip E. Gill and Walter Murray and Michael A. Saunders, "SNOPT Version 7: Software for Large -Scale Nonlinear Programming的用户指南", 。

212,月 2006.

8. Michele Conforti and G{\'e}rard Cornu{'e}jols and Giacomo Zambelli and others, "Integer Programming",Springer , Vol. 271,2014.

9. Juan Pablo Vielma, "Mixed integer linear programming formulation techniques", *Siam Review*, vol. no57,. pp, 1,2015.

上一章 目录 下一章 可访问性

最后修改 2021-6-13.
如何引用这些笔记，使用注释，并给予反馈。

注意：这些是用于在麻省理工学院教授的课程的工作笔记。 它们将在2021年春季学期中被更新。 讲座视频可在YouTube上找到。

上一章 目录 下一章

附录D
优化游戏手册

即将推出.....一个可以使看似不平滑的约束变得平滑、看似不凸的约束变得凸起等的技巧和公式的集合。

上一章 目录 下一章 可访问性

低动能机器人技术

走路、跑步、游泳、飞行和操纵的算法

吕斯-特德雷克

最后修改 2021-6-13.
如何引用这些笔记，使用注释，并给予反馈。

注意：这些是用于在麻省理工学院教授的课程的工作笔记。它们将在2021年春季学期中被更新。讲座视频可在YouTube上找到。

上一章 目录

附录E
杂项

E.1 如何引用这些笔记

谢谢你在你的工作中引用这些笔记。请使用以下引用方式。

Russ Tedrake. *低动能机器人学。步行、跑步、游泳、飞行和操纵的算法（MIT 6.832的课程笔记）。*于[日期]从http://underactuated.mit.edu/ 下载

E.2 注释工具的礼节

我使用注释工具的主要目的是为了主持一个关于文本知识内容的完全开放的对话。然而，它也有一个额外的目的：它是指出我的各种错别字和语法漏洞的一个方便途径。唯一的问题是，如果你高亮了一个错别字，而我几分钟10后就把它改掉了，你的高亮就会永远存在。这最终会污染注释的内容。

有两种可能的解决方案。

- 你可以发表公开的编辑性评论，但必须保证在评论被处理后删除该评论。
- 你可以加入我的 "编辑部 "小组，用这个小组的 "范围 "发表你的编辑评论。

理想情况下，一旦我将评论标记为 "完成"，如果你能删除该评论，我将非常感激。

我高度重视讨论和更正。请继续提供，并感谢你们!

E.3 一些伟大的最终项目

每个学期，学生们都会用这门课的工具完成一个Ønal项目。其中许多项目都很精彩！这里有一个小例子。下面是一个小例子。

2020年春季。

- Bernhard Paus Græsdal的《使用凸形安全区域的四旋翼飞机无碰撞混合积分规划》。
- 查尔斯-道森的《通过轨迹优化的煎饼∞ipping》。
- 动态翱翔》 作者：Lukas Lao Beyer
- 马特-奇诺利和AJ-米勒的《杂技人形》。
- 古田摆的轨迹优化 Samuel Cherna 和 Philip Murzynowski 著

甚至在我们开始在YouTube上发布项目介绍之前，课堂上的一些伟大项目已经变成了完整的出版物。这里有几个例子。

- 推进器-滑块系统的反馈控制。混合动力和欠动力接触动力学的故事
- 利用结构进行基于价值的规划和强化学习
- 在《迷你猎豹》中著名的迷你猎豹背部∞ip。突破动态四足动物控制极限的平台

E.4 请给我反馈！

我对你的反馈非常感兴趣。注释工具是一种机制，但你也可以直接在YouTube讲座上发表评论，或者甚至向承载这些课程笔记的github repo添加问题。

我还创建了这个简单的调查，以收集你的一般意见/反馈。
