arXiv:2302.01413v1 [physics.comp-ph] 2 Feb 2023

使用物理信息机器学习求解二维量子本征值问题

Elliott G. Holliday*, [1] John F. Lindner, [1,2] 和 William L. Ditto[1]
[1] *Nonlinear Artificial Intelligence Laboratory, Physics Department, North Carolina State University, Raleigh, NC 27607, USA*
[2] *Physics Department, The College of Wooster, Wooster, OH 44691, USA*
（日期：2023年2月6日）

受限于不可穿透盒中的粒子，是一种以无限深方势阱建模的典型且可精确求解的一维量子系统。本文使用物理信息神经网络，探索其在二维中的若干无限多种推广，包括受限于矩形、椭圆、三角形和心形盒中的粒子。具体而言，我们将一种无监督学习算法推广用于求出粒子的本征值和本征函数。在训练过程中，神经网络调整其权重和偏置，其中一个参数是能量本征值，因此其输出近似满足薛定谔方程，且本征函数经过归一化并彼此正交。同样的方法也能求解亥姆霍兹方程，得到鼓面上波的谐波和振动模态，或电磁腔的横磁模。相关应用包括动力学台球、量子混沌和拉普拉斯谱。

一、引言

自千禧年之交以来，人工智能深刻影响了我们的文化与生活，从手机到自动驾驶汽车，影响无处不在。科学家近期开始运用机器学习技术，不仅加深我们对周遭世界的理解，也改变了我们处理科学与计算方法的方式，其中包括用于求解描述世界中物理现象的基本微分方程的方法。

以往的研究已使用物理信息神经网络，结合拉格朗日形式和哈密顿形式，求解有序与混沌动力学中的经典物理问题 [1–3]。在更基础的层面上，研究者开发了用于寻找此类动力系统中的对称性 [4], 、守恒定律 [5], 和不变量的方法 [6]。此外，研究还针对热传递 [7], 、不可逆过程 [8], 和耗散系统 [9]等特定问题展开。这些技术甚至已应用于量子系统 [10–13]。本研究使用物理信息神经网络，求解粒子被限制在各种形状的不可穿透平面盒中的量子本征值问题。

我们的工作是对 Jin、Mattheakis 和 Protopapas 研究的扩展，他们 [12, 13], 使用神经网络来求解少数几种系统的一维量子本征值问题。在此，我们将 JMP 算法扩展到二维，并求出具有经典规则或混沌台球动力学的二维区域中带有狄利克雷边界条件的薛"定谔微分方程的本征值和本征函数，其中包括矩形和心形线。

JMP 最重要的特征之一是采用无监督学习，因此在训练过程中不会向神经网络提供微分方程的解。这一特性展示了神经网络寻找问题解的天然能力，即使这些问题可能无法通过其他方法解析地或数值地求解。将这项工作扩展到多维系统，不仅推动了机器学习和物理学的发展，还拓宽了神经网络的实用性，并增加了科学家解决问题的途径。

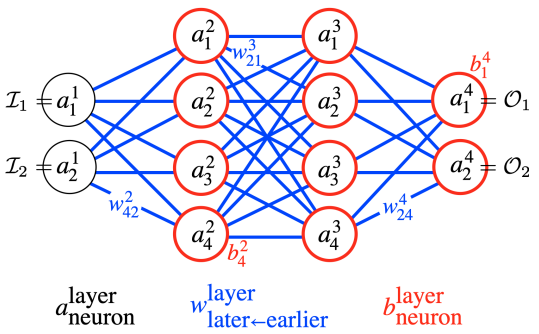
II. 传统神经网络

受哺乳动物大脑的启发，传统的前馈神经网络是由许多称为权重 w^l_{nm} 和偏置 b^l_n的参数决定的嵌套非线性函数，如图 1 的示意图所示。训练通过调整权重和偏置来逼近期望的结果。

给定一个非线性的神经元激活函数，例如$\sigma[z] = \tanh z$, 训练会根据神经元输入递归地更新神经元活动 $a^l_n = \sigma[z^l_n]$

$$z^l_n = \sum_{m=1}^{N^{l-1}} w^l_{nm}\sigma[z^{l-1}_m] + b^l_n, \tag{1}$$

对于第 $2 \le l \le L$ 层的神经元 $1 \le n \le N^l$ ，其中网络的输入和输出分别为 $\mathcal{I}_m = a^1_m$ 和 $\mathcal{O}_m =$



FI图 1. 传统前馈 2:4:4:2 神经网络 网络。

a_m^L。如附录 A 所述，随机梯度下降会调整权重和偏置，以最小化误差（目标、代价、损失）函数，例如

$$L = \sum_{s=1}^{N^L} \frac{1}{2} \left(\mathcal{O}_s - \hat{\mathcal{O}}_s \right)^2, \tag{2}$$

ˆ其中 $\mathcal{O}_s$ 是所需的输出。

III. 方法

A. 一维回顾

我们首先回顾 Jin 等人 [12, 13]针对一维薛定谔方程本征值—本征函数问题提出的形式化方法。神经网络的输入是位置 x 和常数 1，其中常数通过仿射变换转换为能量本征值 λ。如果神经网络的输出是函数 $f(x, \lambda)$，那么本征函数

$$\Psi(x) = \Psi_b + g(x) f(x, \lambda), \tag{3}$$

其中 Ψ_b 是该函数在边界处的值，而 $g(x) = 0$ 位于边界上。

该神经网络的损失函数包含定态一维薛定谔方程，

$$H\Psi(x) = \left(-\frac{\partial^2}{\partial x^2} + V(x) \right) \Psi(x) = E\Psi(x), \tag{4}$$

其中 $V(x)$ 是势能函数，$\Psi(x)$ 可以取实值。构造出 $\Psi(x)$ 后，使用自动微分计算相应的导数。待最小化的损失函数为

$$L = \langle (H\Psi - \lambda\Psi)^2 \rangle_x + L_{\text{reg}}, \tag{5}$$

其中 $\langle \cdot \rangle_x$ 表示对位置 x 求平均，正则化 L_{reg} 包含两个部分，即 L_{norm} 和 L_{ortho}，它们借助物理学指导的直觉，有助于搜索非平凡的高阶本征值和本征函数。

正则化的第一部分基于量子本征函数的归一化性质，即 $\Psi \cdot \Psi < \infty$。通过损失函数强制满足这一性质

$$L_{\text{norm}} = \left(\Psi \cdot \Psi - \frac{M}{\Delta l} \right)^2, \tag{6}$$

其中 M 是归一化常数，Δl 是势函数的长度尺度。损失函数的这一部分会抑制神经网络找到平凡恒零解。正则化的第二部分基于量子本征函数的正交性，即 $\Psi_1 \cdot \Psi_2 = 0$。通过损失函数强制满足这一性质

$$L_{\text{ortho}} = \Psi_s \cdot \Psi, \tag{7}$$

其中 Ψ_s 是此前已学习的本征函数之和，而 Ψ 是当前正在学习的本征函数。

B. 二维扩展

接下来，我们将一维 JMP算法扩展到二维，并稍作重新表述。该网络包含两个隐藏层，每层由采用正弦激活函数 $\sigma(z) = \sin z$ 的神经元组成。输入为位置 $\{x, y\}$ 和常数 1；与图 2 中一样，常数 1 通过可调权重 W_{11}^1转换为能量本征值 $E = W_{11}^1$。输出是不完备本征函数 $\psi_E(x, y)$，它与一个函数 $B(x, y)$ 相乘。该函数在盒子的周界 $\partial\Omega$ 上为零，以满足边界条件并生成完备本征函数

$$\Psi_E(x, y) = B(x, y)\, \psi_E(x, y). \tag{8}$$

给定第一本征函数 Ψ_1 时，求第二本征函数 Ψ_2 的损失

$$\begin{aligned} L &= L_D + L_N + L_O \\ &= ||H\Psi_2 - E_2\Psi_2||^2 \\ &+ \lambda_N (||\Psi_2|| - 1)^2 \\ &+ \lambda_O \langle \Psi_2 | \Psi_1 \rangle, \end{aligned} \tag{9}$$

其中哈密顿算符

$$H = -\frac{\partial^2}{\partial x^2} - \frac{\partial^2}{\partial y^2} + V(x, y), \tag{10}$$

以及标量积

$$\langle \Psi | \Phi \rangle = \int_{-\infty}^{\infty} dx \int_{-\infty}^{\infty} dy\, \Psi(x, y)^* \Phi(x, y), \tag{11}$$

以及范数平方

$$||\Psi||^2 = \langle \Psi | \Psi \rangle = \int_{-\infty}^{\infty} dx \int_{-\infty}^{\infty} dy\, |\Psi(x, y)|^2. \tag{12}$$

附录 B 讨论了归一化损失的不同形式。我们假设本征函数为实数，即 $\Psi^* = \Psi$ 和 $\Phi^* = \Phi$，并将积分近似为求和，因此

$$\begin{aligned} \langle \Psi | \Phi \rangle &\approx \sum_{m=0}^{M} \sum_{n=0}^{N} \delta x\, \delta y\, \Psi(x_m, y_n) \Phi(x_m, y_n) \\ &\approx \Delta x\, \Delta y\, \overline{\Psi_{mn} \Phi_{mn}} \end{aligned} \tag{13}$$

and

$$\begin{aligned} ||\Psi||^2 &\approx \sum_{m=0}^{M} \sum_{n=0}^{N} \delta x\, \delta y\, \Psi(x_m, y_n)^2 \\ &\approx \Delta x\, \Delta y\, \overline{\Psi_{mn}^2} \end{aligned} \tag{14}$$

其中，对于势阱 $\Omega \subset [0, \Delta x] \times [0, \Delta y]$，$x_m = m\,\delta x = m\,\Delta x/M$ 和 $y_n = n\,\delta y = m\,\Delta y/N$，上方的横线表示平均值。

最小化微分方程损失 L_D 可确保满足薛定谔方程，最小化归一化损失 L_N 可避免平凡零解，而最小化正交损失 L_O 则有助于获得独立解。学习会持续进行，直到总损失 L、微分方程损失 L_D 以及 其变化率都足够小。附录 C 讨论了实现细节。

IV. 示例

我们考虑形式如下的势能函数

$$V(x,y)=\begin{cases}0, & x,y\in\Omega,\\ \infty, & x,y\notin\Omega,\end{cases} \tag{15}$$

其中 Ω 是势阱的内部， $\partial\Omega$是其边界。我们的研究包括被限制在矩形、椭圆、三角形和心形势阱中的粒子。矩形的本征函数包含正弦函数，椭圆的本征函数包含马丢函数 [14, 15], ，而三角形和心形的本征函数则更为复杂。

从经典台球的角度来看，矩形盒和椭圆形盒是非遍历且可积的，而心形盒（其极坐标边界为 $r=1-\delta\sin\theta$）在凸（$0\le\delta<1/2$）时具有混合相空间，在凹（$1/2<\delta\le1$）时则是遍历、混合且混沌的。具有无理角（即 π 的无理倍数）的三角形盒是遍历且混合的，但并不混沌；不过，具有一个或多个有理角的三角形盒甚至可能不是遍历的 [17, 18]。（从谱分析的角度来看，在狄利克雷边界条件固定时，已知其拉普拉斯谱的三角形只有等边三角形 $60°-60°-60°$、等腰直角三角形 $45°-45°-90°$ 和半等边三角形 $30°-60°-90°$ [19]。）

我们预计，可积的矩形和椭圆形盒中的能量本征值间距服从泊松统计，而混沌心形盒中的能量本征值间距则服从高斯正交系综（GOE）统计 [20, 21], ，三角形的谱统计则介于两者之间 [17]。

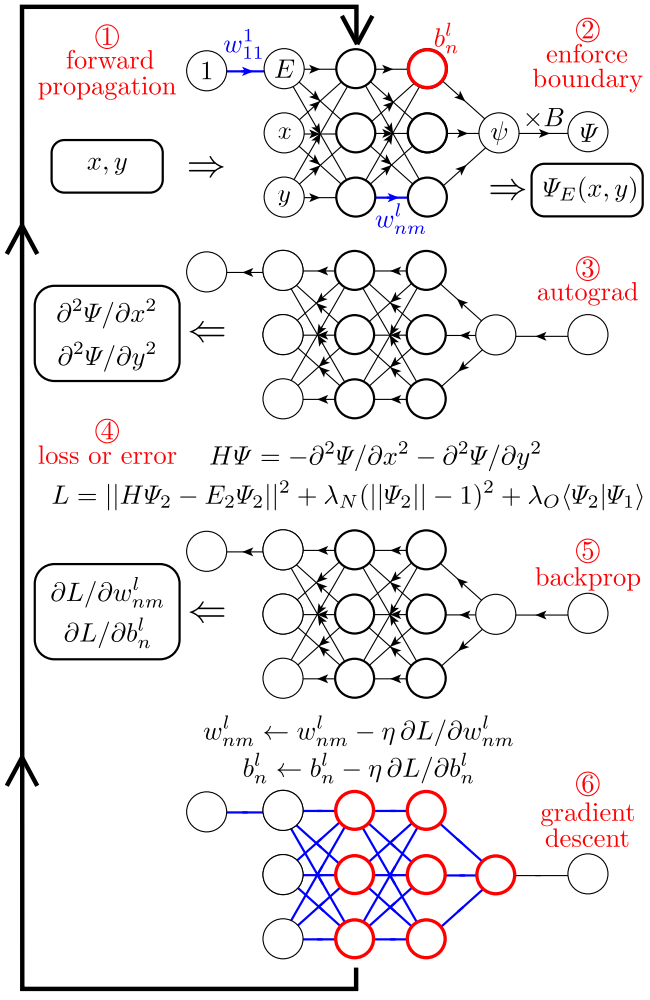


图 2. 给定第一本征函数 Ψ_2 后，神经网络通过梯度下降求得第二本征函数 Ψ_1的一步过程。箭头表示权重 w^l_{nm} ，圆圈表示偏置 b^l_n。实际计算中，大部分运算涉及反向模式自动微分。

V. 结果

我们通过构建一个全连接前馈神经网络来开展研究，该网络包含 $1+2$ 个输入、2 个隐藏层（每层包含 150 到 200 个神经元，具体取决于势的复杂程度）以及 1 个输出。为了训练神经网络，我们在盒内生成 100 个随机 $\{x,y\}$ 点，并按照图 2 中概述的算法将它们输入网络。我们重复约 10^5 个训练循环或轮次。

如图 3 所示，神经网络调整其权重和偏置，使其收敛至基态能量，并停留在该能量平台，直到将正交项 L_O 加入损失函数中。加入正交项后，

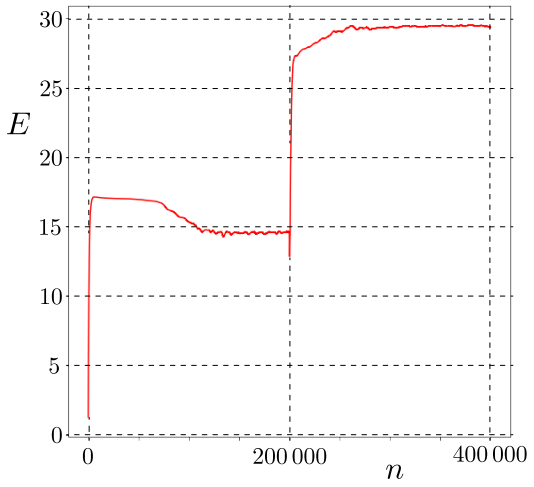
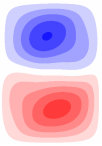
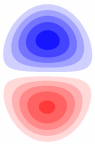
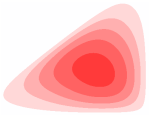
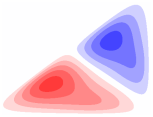


图 3. 矩形盒的能量 E 随训练步数 s 的变化。我们通过寻找能量平台来判断神经网络是否已找到能量本征值。第 200 000 个训练步时出现的跃升，是由于引入了正交损失 L_O，它促使神经网络寻找下一个本征值。

表 I。示例摘要。边界函数 $B(x,\ y)$、在 Mathematica [16], 中计算得到的参考本征值 E_n^*、神经网络本征值近似值 E_n、相对误差 $\Delta E_n/E_n$、神经网络特征函数近似 $\Psi_n(x,\ y)$。红、白、蓝配色表示正值、零值和负值。

边界函数 B	参考值 E_n	神经网络 E_n	误差 $\Delta E_n/E_n$	神经网络 Ψ_1	神经网络 Ψ_2
矩形 $\frac{x}{a}\left(1-\frac{x}{a}\right)\frac{y}{b}\left(1-\frac{y}{b}\right)$ 其中 $a=1$ 和 $b=\sqrt{2}$	14.8 29.6	14.5 ± 0.1 29.4 ± 0.1	-2.0% -0.68%		
椭圆 $1-\left(\frac{x}{a}\right)^2-\left(\frac{y}{b}\right)^2$ 其中 $a=1$ 和 $b=\sqrt{2}$	4.32 9.13	4.32 ± 0.01 9.11 ± 0.08	0.00% -0.22%		
三角形 $\left(1-\frac{x}{a}\right)\left(\frac{y}{b}-\frac{x}{a}\right)\frac{y}{b}$ 正切 θ 其中 $a=4$、 $b=4$和 $\theta=\pi/\sqrt{23}$	4.09 8.00	4.06 ± 0.01 7.98 ± 0.03	-0.73% -0.25%		
心形线 $r=1-\delta$ 正弦 $\theta\Rightarrow$ $\left(\frac{x}{a}\right)^2+\frac{y}{b}\left(\frac{y}{b}+\delta\right)-\sqrt{\left(\frac{x}{a}\right)^2+\left(\frac{y}{b}\right)^2}$ 其中 $a=1$, $b=1$, 且 $\delta=1$	4.05 9.12	4.17 ± 0.15 9.13 ± 0.24	$+3.0\%$ $+0.11\%$		

神经网络便离开基态能量平台，寻找与第一激发态相关的能量。

表 I 总结了结果。参考能量通过 Mathematica [16] 数值地计算得出（并对矩形进行了精确核验）。能量本征值的不确定性是能量平台振荡衰减至其平均值时的波动；相对误差 $\Delta E/E$ 为估算能量减去参考能量，再除以参考能量。本征函数的色彩调色板从完全饱和的红色（对应 $\Psi>0$）渐变至完全饱和的蓝色（对应 $\Psi<0$），并经过完全不饱和的白色（对应 $\Psi=0$）。

较小的盒子具有更高的能态，因为动量 $p\propto 1/\lambda$ 意味着能量 $E=p^2/2m\propto 1/\lambda^2$，其中$\lambda$ 是量子化波长。因此，可以调整表 I 中的缩放参数 a，b，使能量本征值处于方便的数值范围内。

实现二维 JMP 算法的神经网络，成功近似了粒子被限制在各种形状的盒子中时的基态和第一激发态能量本征值及本征函数，且无需假设或施加盒子的对称性。通过在损失函数中添加更多正交项，可以类似地获得更高阶激发态。进一步训练可提高精度。最难处理、也最令人印象深刻的是心形线形或夹 pinch 圆形盒，其第二和第三态几乎简并，其中顶部的尖角打破了简并，使具有水平节点的本征函数能量略高于具有垂直节点的本征函数。

VI. 其他应用

相关应用包括声学和电磁腔体以及拉普拉斯谱。含时无关的薛定谔方程（式 4）可写为

$$(-\nabla^2+V)\Psi=E\Psi. \tag{16}$$

在硬壁无限深势阱中，$V=0$ 和

$$\nabla^2\Psi=-E\Psi. \tag{17}$$

这与亥姆霍兹方程相同

$$\nabla^2 f=-k^2 f, \tag{18}$$

它可以描述二维中固定边界膜上的波 [22]，以及三维导电腔体中的电磁波 [23, 24]。拉普拉斯方程是一个特例

$$\nabla^2 f=0, \tag{19}$$

这是平面区域在狄利克雷边界条件下拉普拉斯本征值数学问题的核心内容 [19]。

VII. 结论

我们已经证明，将JMP算法扩展到二维后，神经网络能够求解含时独立的薛定谔方程，并为经典规则和不规则的台球系统求得量子能量本征值和本征函数。这种能力再次展示了物理学引导的机器学习如何驾驭同时呈现有序与混沌特征的动力系统 [1]。

这一成果证明了该概念：在损失函数中融入物理学直觉的简单前馈神经网络，能够求解复杂的本征值问题，即使成熟的先进数值方法目前速度更快或精度更高也是如此。二维 JMP 神经网络具有很大潜力。未来的工作包括利用空间对称性减少训练点数量，并将方法推广到连续势阱和三维势阱。

致谢

本研究得到了美国海军研究办公室资助（项目编号 N00014-16-1-3066）以及 United Therapeutics Corporation 的捐赠支持。

附录 A：梯度下降

1. 动力学类比

神经网络权重 w 的梯度下降，就像质量为 m 、位于位置 x 的点粒子，在势能面 $V(x)$上以黏滞系数 γ 滑动。根据牛顿定律，

$$m\ddot{x} = F_x = -\frac{dV}{dx} - \gamma\dot{x}, \tag{A1}$$

其中上方的点表示对时间求导。当黏滞系数 $|m\ddot{x}| \ll |\gamma\dot{x}|$ 较大时，且

$$\gamma\dot{x} \sim -\frac{dV}{dx}, \tag{A2}$$

因此速度

$$\dot{x} \sim -\frac{1}{\gamma}\frac{dV}{dx}. \tag{A3}$$

位置的演化类似于欧拉更新

$$x \leftarrow x + dx = x + \dot{x}\,dt \sim x - \frac{1}{\gamma}\frac{dV}{dx}dt. \tag{A4}$$

对于位置 $x = w$、高度 $V = L$ 和学习率$\eta = dt/\gamma$，神经网络权重的演变方式如下

$$w \leftarrow w - \eta\frac{dL}{dw}, \tag{A5}$$

偏置的变化方式也类似。增大学习率会让损失面变得更“滑”，而减小学习率则会让损失面变得更“黏”；采用可变学习率可能会加快梯度下降，使其达到全局最小值。通过用噪声冲击滑动的粒子，模拟随机 梯度下降。

2. 牛顿法

另一种方法是根据牛顿-拉夫森方法，延长切线至截距处并向前迈步，通过寻找导数 $L'(w)$ 的根来求损失函数 $L(w)$ 的极小值

$$w \leftarrow w - \frac{L'(w)}{L''(w)}. \tag{A6}$$

如果 w_m 处的极小值近似为二次函数，则

$$L(w) = \frac{1}{2}a(w - w_m)^2 + c, \tag{A7}$$

那么在其附近，牛顿法可化为

$$w \leftarrow w - \frac{1}{a}L'(w) = w - \eta\frac{dL}{dw}, \tag{A8}$$

其中，学习率 $\eta = 1/a = 1/L''(w)$与曲率成反比。

附录 B：归一化损失

对于归一化损失，Jin 等人 [13] 提出

$$\begin{aligned} L_N &= (\langle\Phi|\Phi\rangle - M/\Delta x)^2 \\ &= c\,(||\Psi||^2 - 1)^2, \end{aligned} \tag{B1}$$

其中 $c = (M/\Delta x)^2$ 以及 $|\Psi\rangle = |\Phi\rangle/\sqrt{M/\Delta x}$。然而，另一种损失为

$$L_N = c\,(||\Psi|| - 1)^2. \tag{B2}$$

后者可以说更简单，而相关函数

$$L_N = c\,(||\Psi||^2 - 1) \tag{B3}$$

这是有问题的，因为它可能为正，也可能为负。

附录 C：实现细节

图 4 中的 Python 示例代码实现了一个简单的神经网络，该网络使用 Sigmoid 激活函数来学习一维盒子中粒子的基态能量本征值和本征函数 $\Omega = [0, 1]$。我们的 PyTorch 机器学习库实现全程使用张量（数字的多维矩形数组）。ClassNet（第 10–32 行）定义了网络架构（第 14–19 行），实现了前向传播（第 21–32 行），并强制满足狄利克雷边界条件（第 31 行）。实例化类对象 net 并初始化随机梯度下降优化器（第 34–35 行）后，变量 x 是盒子内等间距位置构成的列表或列数组，自动微分会跟踪其运算（第 37–38 行）。

for 循环（第 40–54 行）负责神经网络训练：首先打乱 x 值（第 41 行），然后请求 net 对象返回最新的本征函数和能量本征值近似值（第 42 行）。grad 函数调用自动微分来计算拉普拉斯算子（第 44–46 行）。.pow() 和 .mean() 方法用于计算损失函数（第 48–50 行）。

.backward() 方法也会调用自动微分，并计算损失相对于权重和偏置的梯度，然后由优化器更新这些权重和偏置（第 52–54 行）。更一般地说，.backward() 方法会计算计算图中所有张量的 .grad 属性，这些张量的 requires grad = True；在该计算图中，损失是最终叶节点，输入是根节点。随后，优化器会遍历初始化时接收的权重和偏置参数列表；只要某个张量的 requires grad = True，就会从中减去其 .grad 属性中存储的梯度值（乘以学习率）。

最终的能量本征值和本征函数以数值形式提取并打印出来（第 56—58 行）。这个工作示例返回的基态能量与精确值相差约 1%。

```
1  import torch                    # machine-learning library
2  torch.manual_seed(0)            # eliminate nondeterminism
3
4  n_neurons = 100         # number of neurons per hidden layer
5  n_epochs = 1000 + 1     # number of times net sees shuffled x
6  n_steps = 1000 + 1      # number of sampling steps in x
7  learning_rate = 0.01    # for gradient descent
8  lambda_N = 6.0          # normalization-loss weight
9
10 class ClassNet(torch.nn.Module):         # subclass neural network
11     def __init__(self, n_neurons):       # constructor method
12         super(ClassNet,self).__init__()
13
14         self.activ = torch.nn.Sigmoid()  # activation function
15
16         self.lin_0 = torch.nn.Linear(1,1,False) # E = w * 1 + 0
17         self.lin_1 = torch.nn.Linear(2, n_neurons)
18         self.lin_2 = torch.nn.Linear(n_neurons, n_neurons)
19         self.lin_3 = torch.nn.Linear(n_neurons, 1)
20
21     def forward(self, x):         # forward propagation method
22         E = self.lin_0(torch.ones_like(x))  # create 1s
23         x_E = torch.cat((x, E),1) # interleave x & E inputs
24
25         i_o  = self.lin_1(x_E)    # process input to output
26         i_o  = self.activ(i_o)
27         i_o  = self.lin_2(i_o)
28         i_o  = self.activ(i_o)
29         psi_ = self.lin_3(i_o)
30
31         psi = x * (1 - x) * psi_   # enforce boundary conditions
32         return (psi, E)            # approx wavefunction & energy
33
34 net = ClassNet(n_neurons)   # invokes object_net.init(x)
35 optimizer = torch.optim.SGD(net.parameters(), lr = learning_rate)
36
37 x0 = torch.linspace(0,1,n_steps).reshape(-1,1) # positions column
38 x0.requires_grad = True # so autograd tracks operations on x0
39
40 for n in range(n_epochs):           # training
41     x = x0[torch.randperm(n_steps)] # shuffle for stochasticity
42     (psi, E) = net(x)               # invokes object_net.forward(x)
43
44     s = torch.ones_like(psi)        # 1s -> shape of derivatives
45     psi_x = torch.autograd.grad(psi, x, s, create_graph=True)[0]
46     psi_xx = torch.autograd.grad(psi_x, x, s, retain_graph=True)[0]
47
48     L_D = (- psi_xx - E * psi).pow(2).mean() # differential eq loss
49     L_N = (psi.pow(2).mean() - 1).pow(2)     # normalization loss
50     L = L_D + lambda_N * L_N                 # total loss
51
52     optimizer.zero_grad()       # zero accumulated gradients
53     L.backward()                # compute gradients dL/w, dL/db
54     optimizer.step()            # update parameters w & b
55
56 (psi_final, E_final) = net(x0)              # get final results
57 print("E =", E_final[0].item())             # energy eigenvalue
58 print("ψ =", psi_final.flatten().tolist())  # eigenfunction
```

图 4. 用于学习建模一维盒子中粒子的神经网络 Python 代码示例 $\Omega = [0, 1]$。返回的基态能量与精确值相差约 1%。

[1] Anshul Choudhary、John F. Lindner、Elliott G. Holliday、Scott T. Miller、Sudeshna Sinha 和 William L. Ditto。《物理增强神经网络学习有序与混沌》。*Physical Review E*，101，2020 年 6 月。[2]Marc Finzi、Ke Alexander Wang 和 Andrew Gordon Wilson。通过显式约束简化哈密顿神经网络和拉格朗日神经网络，2020。arxiv.org/abs/2010.13581。[3] Marios Mattheakis、David Sondak、Akshunna S. Dogra 和 Pavlos Protopapas。用于求解

运动方程的哈密顿神经网络。 *Phys. Rev. E*，105:065305，2022年6月。[4] Roberto Bondesan 和 Austen Lamacraft。学习经典可积系统的对称性，2019。arxiv.org/abs/1906.04645。[5] Ziming Liu 和 Max Tegmark。从轨迹中学习守恒定律。 *Physical Review Letters*，126，2021，。[6] Sebastian J. Wetzel、Roger G. Melko、Joseph Scott、Maysum Panju 和 Vijay Ganesh。通过解读孪生神经网络发现对称不变量和守恒量，2020。arxiv.org/abs/2003.04299。[7] Shengze Cai、Zhiche ng Wang、Sifan Wang、Paris Perdikaris 和 George Em Karniadakis。用于热传递问题的物理信息神经网络。 *Journal of Heat Transfer*，143，6，2021。[8] Kookjin Lee、Nathaniel A. Trask 和 Panos Stinis。通过机器学习保结构括号预测不可逆过程，2021。arxiv.org/abs/2106.12619。[9] Yaofeng Desmond Zhong、Biswadip Dey 和 Amit Chakraborty。耗散型 SymODEN：将含耗散和控制的哈密顿动力学编码到深度学习中，2020。arxiv.org/abs/2002.08860。[10] Huaixin Cao、Feilong Cao 和 Dianhui Wang。量子人工神经网络及其应用 q。 *Information Sciences* 290 1 6, 2015 ，：–。

[11] M. Raissi、P. Perdikaris 和 G. E. Karniadakis。物理信息神经网络：用于求解涉及非线性偏微分方程的正问题与逆问题的深度学习框架。 *Journal of Computational Physics* 378 686 707, 2 2019 ，：–。

[12] Henry Jin、Marios Mattheakis 和 Pavlos Protopapas。用于量子本征值问题的无监督神经网络。载于 *2020 NeurIPS Workshop on Machine Learning and the Physical Sciences* 。

NeurIPS，NeurIPS，2020。[13] Henry Jin、Marios Mattheakis 和 Pavlos Protopapas。

用于量子本征值问题的物理信息神经网络。载于 *IJCNN at IEEE World Congress on Computational Intelligence* 2022 ，。

[14] ´Emile Mathieu。《椭圆形膜振动运动论》（Dissertation on the vibratory movement of an elliptical membrane）。 *Journal de Mathématiques Pures et Appliquées* 13 137 203, 1868 ，：–。

[15] S Chakraborty、J K Bhattacharjee 和 S P Khastgir。二维不规则边界上的一个本征值问题。 *Journal of Physics A: Mathematical and Theoretical* 42 19 195301, 2009 ，（）：

2022 年 4 月。[16] Wolfram Research, Inc. Mathematica, 13.1 版。伊利诺伊州尚佩恩，2022 年。[17] ˇCrt Lozej、Giulio Casati 和 Tomaˇz Prosen。三角形台球中的量子混沌。 *Phys. Rev. Research*，4 2022 013138，：2022 年 2 月。[18] Katerina Zahradova、Julia Slipantschuk、Oscar F. Bandt-low 和 Wolfram Just。对称性对三角形台球遍历性质的影响。 *Phys. Rev. E*，105 012201，2022：L Jan。[19] Brian J. McCartin。关于在狄利克雷或诺伊曼边界条件下具有拉普拉斯算子三角本征函数的多边形区域。 *Applied Mathematical Sciences* 2 58 2891 2901, 2008 ，（）：–。

[20] G. Casati、F. Valz-Gris 和 I. Guarnieri。非可积系统量子化与谱的统计理论之间的联系。 *Lettere al Nuovo Cimento (1971-1985)* 28 8 279 282, 1980 ，（）：–。

[21] O. Bohigas、M. J. Giannoni 和 C. Schmit。混沌量子谱的特征及能级涨落定律的普适性。 *Phys. Rev. Lett.*，52:1–4，1984 年 1 月。[22] J. W. S. Rayleigh。 *Vibrations of membranes*。Dover，1945 年。[23] H.-J. St¨ockmann 和 J. Stein。通过微波吸收研究台球中的“量子”混沌。 *Phys. Rev. Lett.*，64 2215 2218，1990：2022 年 5 月。[24] Jens U N¨ockel 和 A Douglas Stone。非对称谐振光学腔中的射线与波混沌。 *Nature*，385 6611 45 47，1997（）：–。