终于生活都回归正轨了，准备要开学了，接着上次的SSM接着补充。

Day7: The State Space Model (SSM)-下

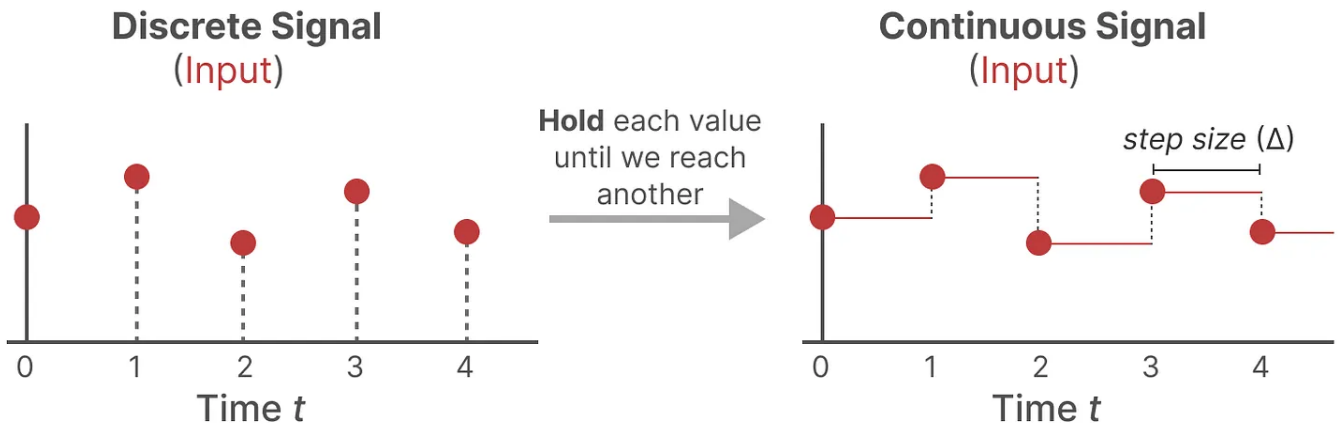
我认为，从最开始的SSM可以看到它具有以下特点：

- 用于预测连续的连续的变量，通过引入可能的中间隐藏状态，算一个状态到另一个状态的概率。因此，SSM和其他深度学习方法在核心思想上都是利用其他变量增强模型的表达能力，更SSM侧重于系统内部状态的动态演化和因果关系。
- 其中，在SSM的观测方程中的`C`控制了反应的强度，`A`对隐藏状态起绝对的控制作用。

但GPU只能计算离散的数据，成为一种新的网络框架除了找到`A`并进行控制，还要使用数学知识将离散的数据序列化。

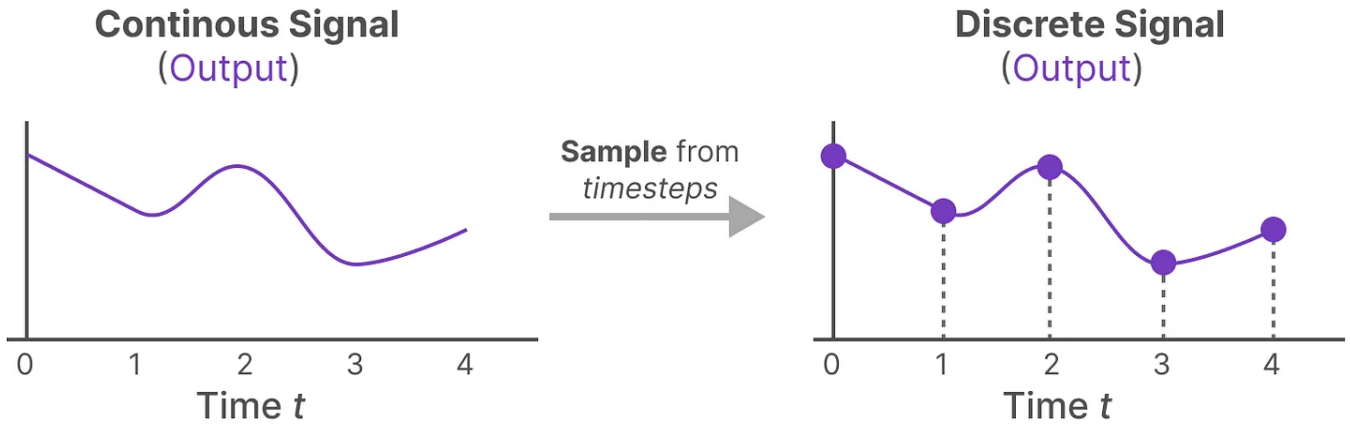
1. 离散数据的连续化

可以利用**零阶保持技术**处理离散的数据：

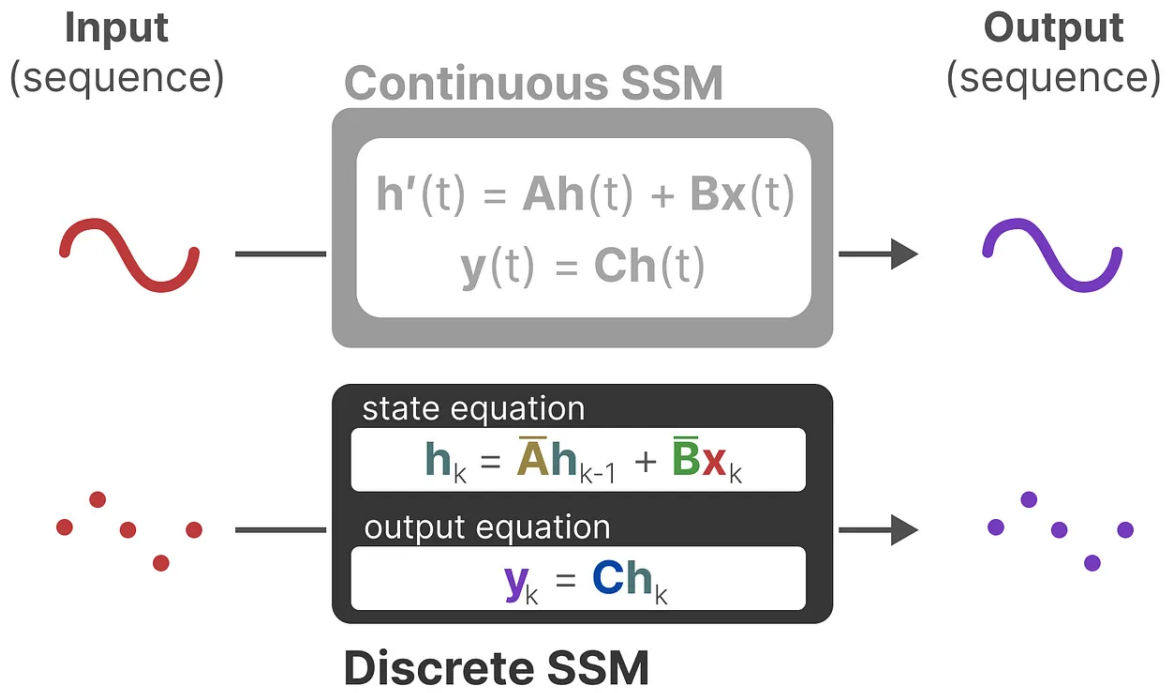


首先，每次收到离散信号时，我们都会保留其值，直到收到新的离散信号，如此操作导致的结果就是创建了 SSM 可以使用的连续信号　保持该值的时间由一个新的可学习参数表示，称为步长(siz)——Δ，它代表输入的阶段性保持(resolution)

有了连续的输入信号后，便可以生成连续的输出，并且仅根据输入的时间步长对值进行采样



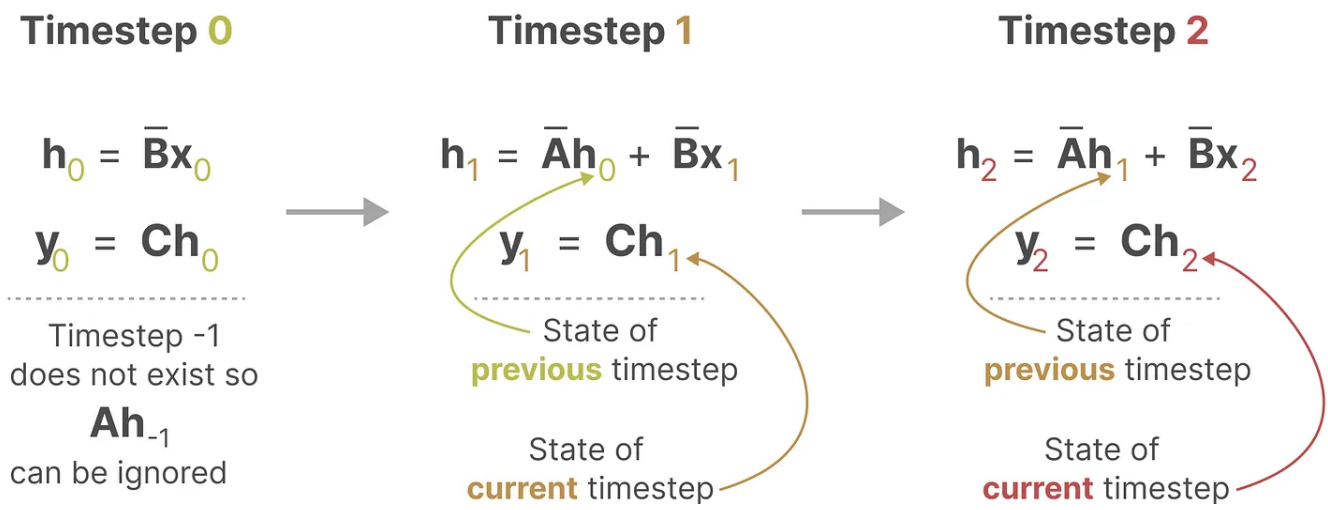
因此，我们可以得到（做了零阶保持的在对应变量上面加了个横杠）：



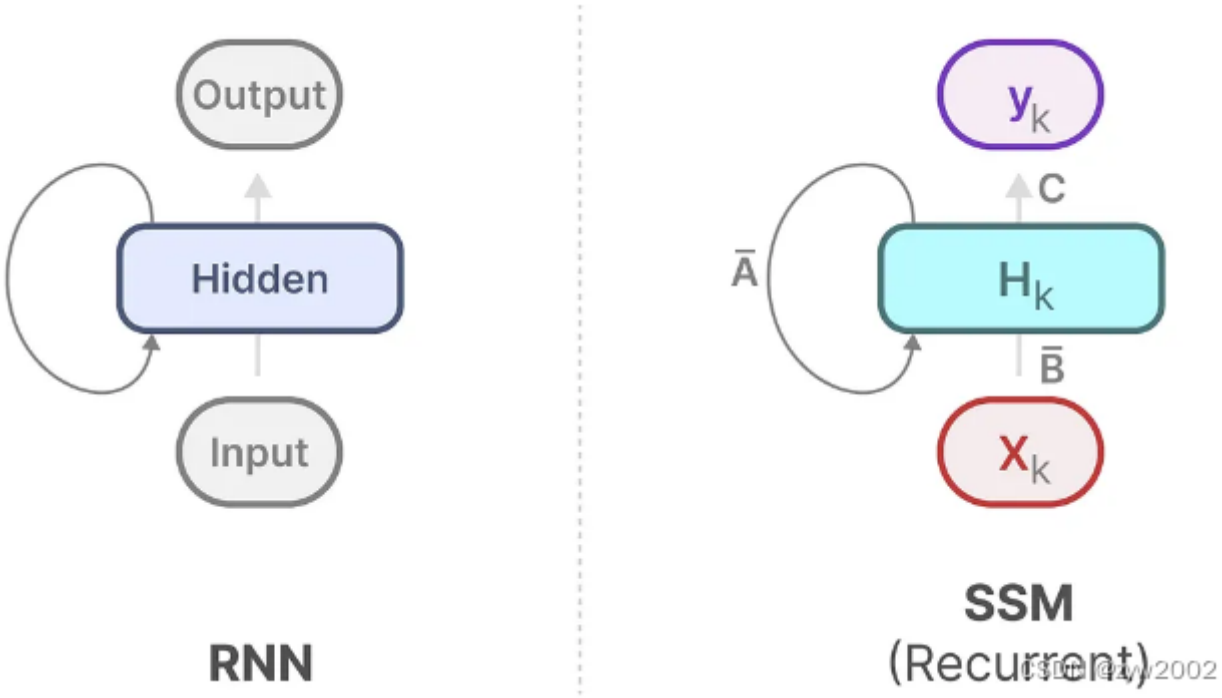
此时，函数从 `x(t) -> y(t)` 变成了 x_k -> y_k 。离散化的SSM可以用用离散时间步长重新表述问题。但我们在保存时，仍然保存矩阵A的连续形式(而非离散化版本)，只是在训练过程中，连续表示被离散化。

2. 更新隐藏状态

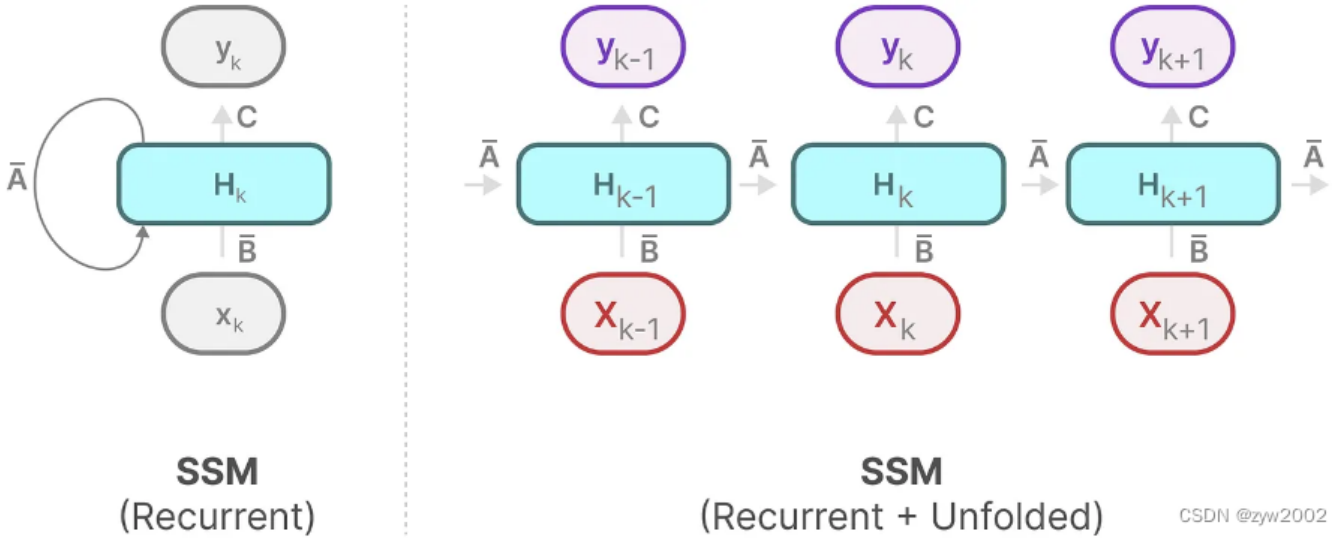
在每个时间步，都会涉及到隐藏状态的更新。在每个时间步长，我们计算当前输入(Bx_k)如何影响前一个状态(Ah_{k-1})，然后计算预测输出(Ch_k)。



此时，我们发现这时的结构和RNN十分相似（隐藏层的输入不仅包括输入层的输出还包括上一时刻隐藏层的输出）：

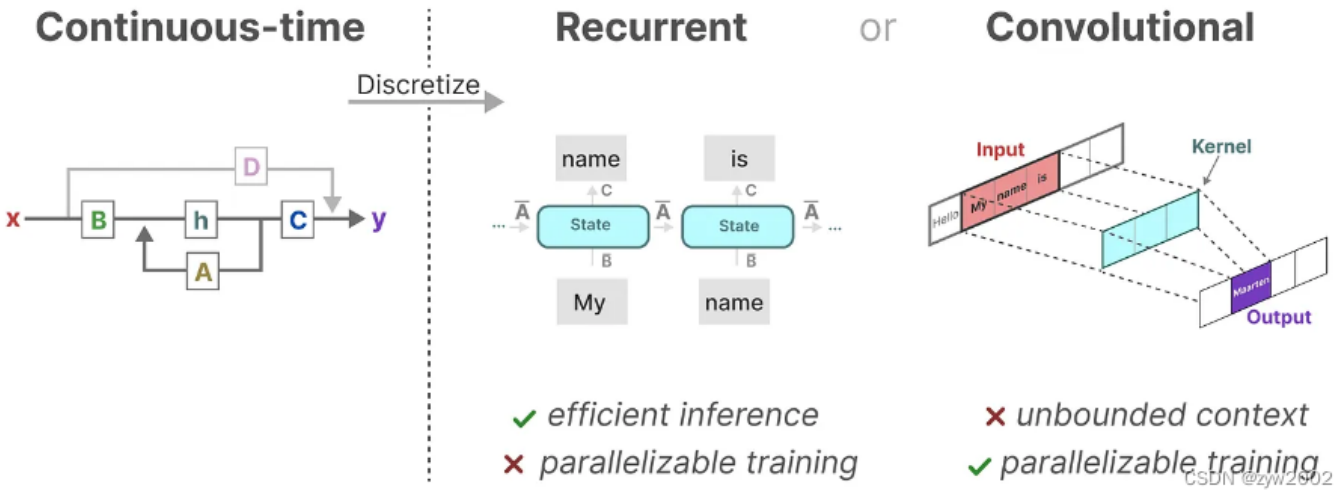


进一步的，我们将这种循环的SSM结构展开，进行可视化：

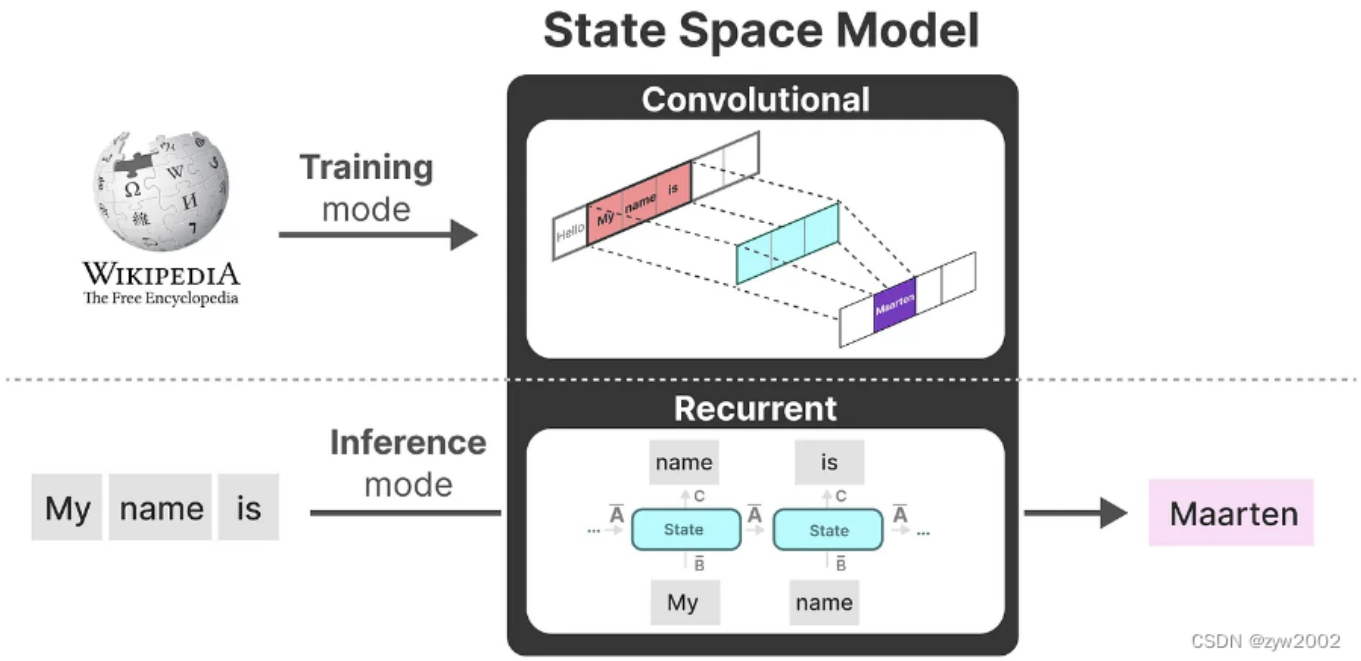


3. LSSL的设计思路

SSM表示为卷积的一个主要好处是它可以像卷积神经网络CNN一样进行并行训练。然而，由于内核大小固定，它们的推理不如RNN那样快速（这里不展开讨论）。



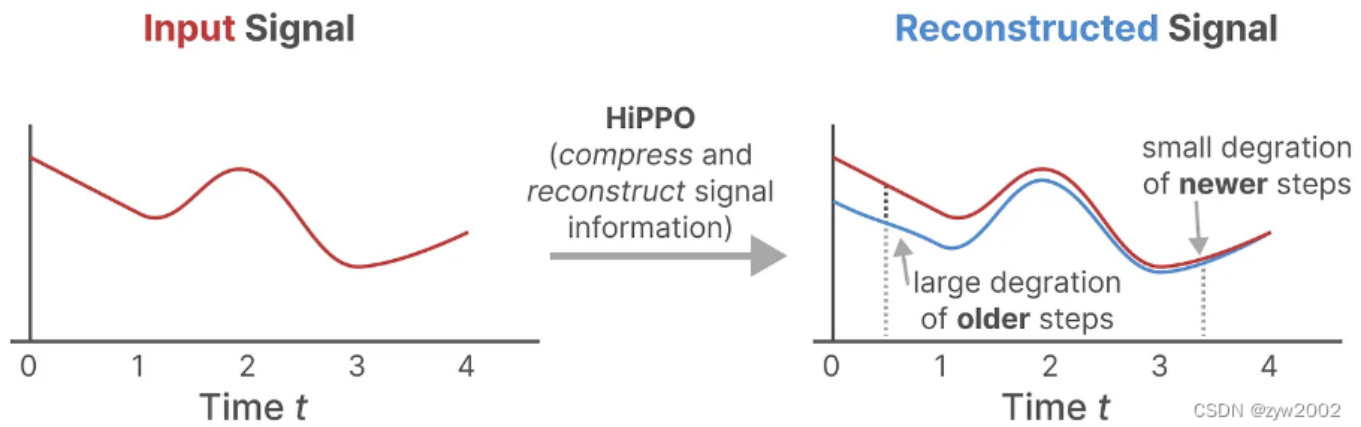
作为从输入信号到输出信号的参数化映射，SSMs可以当做是RNN与CNN的结合，即推理用RNN结构，训练用CNN结构。



LSSL的一个重要特性是线性时间不变（Linear Time Invariance, LTI）： LTI声明SSM参数A、B和C对于所有时间步都是固定的。这意味着矩阵A、B和C对于SSM生成的每个token都是相同的。换句话说，无论你给SSM什么序列，A、B和C的值都保持不变。我们有一个不感知内容(not content-aware)的静态表示。

3. 如何创建A来保留尽可能的长距离信息

由于和RNN相似的推理方式，如何选择和更新A来保留尽可能的长距离信息是主要问题。这里使用的是High-order Polynomial Projection Operators (HiPPO) 。HiPPO试图将它迄今为止看到的所有输入信号压缩为一个系数向量。



HiPPO使用矩阵A来构建状态表示，可以很好地捕获最近的token并衰减旧的token。其公式可以表示为：

$$(\textbf{HiPPO Matrix}) \qquad \boldsymbol{A}_{nk} = -\begin{cases} (2n+1)^{1/2}(2k+1)^{1/2} & \text{if } n > k \\ n+1 & \text{if } n = k \\ 0 & \text{if } n < k \end{cases} \tag{2}$$

HiPPO通过一系列的矩阵更新（或对应的微分方程），将新的输入流投影到多项式系数上，保持对历史输入在多项式基中的近似表示。每一步更新并不是将历史信息全部覆盖，而是调整多项式系数，使它们在最小化误差的意义下继续代表到目前为止的整个输入序列，这就是它"记住"历史的核心机制。

然后将HiPPO应用于我们之前看到的递归和卷积表示，以处理长程依赖关系。其结果是序列的结构化状态空间(Structured State Space for Sequences, S4)，这是一类可以有效处理长序列的SSM。

S4主要包括以下三个部分：

- 状态空间模型
- HiPPO用于处理远程依赖
- 用于创建循环和卷积表示的离散化