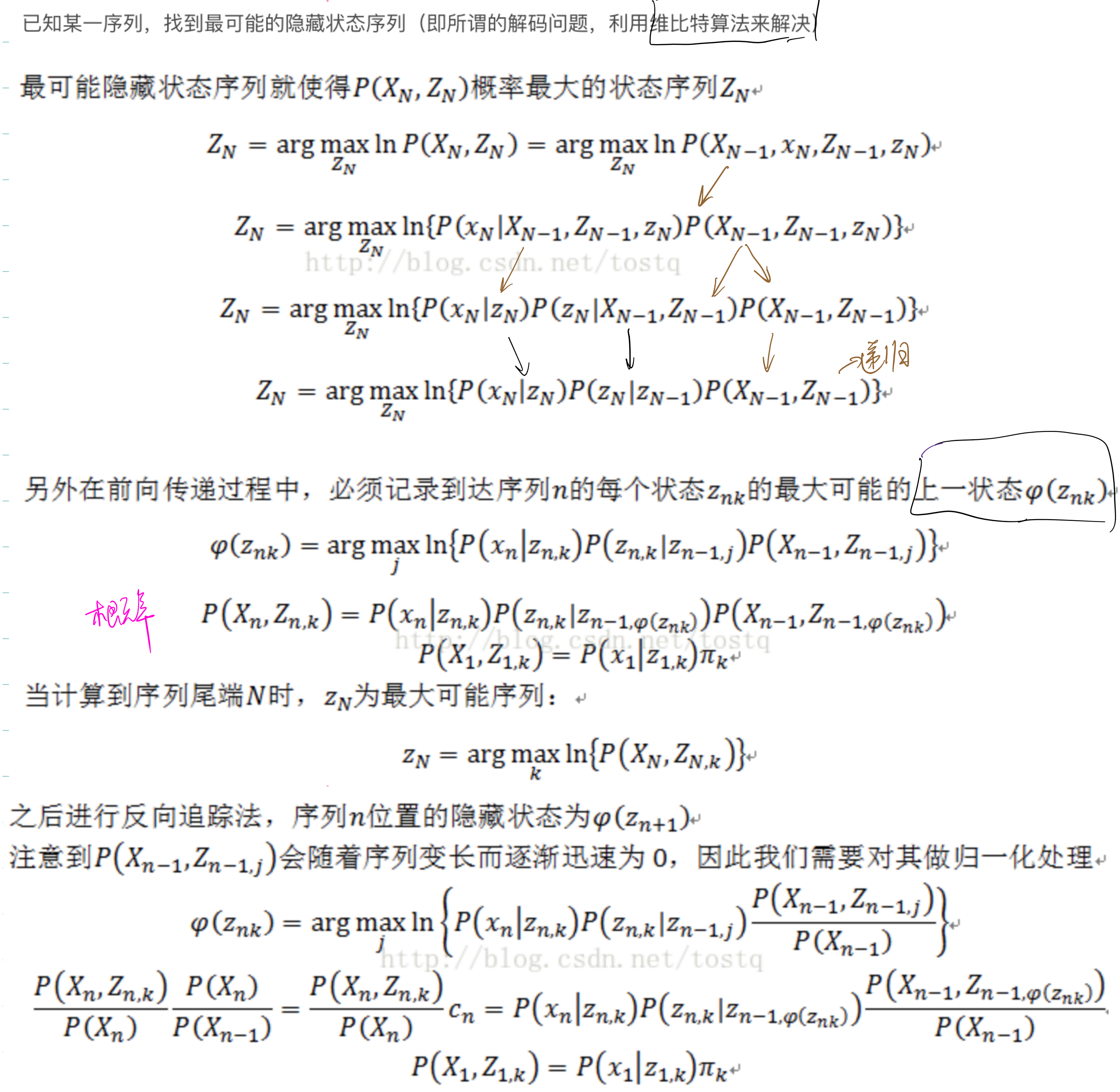
已知某一序列，找到最可能的隐藏状态序列



已知某一序列，找到最可能的隐藏状态序列（即所谓的解码问题，利用维比特算法来解决）

最可能隐藏状态序列就使得$P(X_N, Z_N)$概率最大的状态序列Z_N

$$Z_N = \arg\max_{Z_N} \ln P(X_N, Z_N) = \arg\max_{Z_N} \ln P(X_{N-1}, x_N, Z_{N-1}, z_N)$$

$$Z_N = \arg\max_{Z_N} \ln\{P(x_N|X_{N-1}, Z_{N-1}, z_N)P(X_{N-1}, Z_{N-1}, z_N)\}$$

$$Z_N = \arg\max_{Z_N} \ln\{P(x_N|z_N)P(z_N|X_{N-1}, Z_{N-1})P(X_{N-1}, Z_{N-1})\}$$

递归

$$Z_N = \arg\max_{Z_N} \ln\{P(x_N|z_N)P(z_N|z_{N-1})P(X_{N-1}, Z_{N-1})\}$$

另外在前向传递过程中，必须记录到达序列n的每个状态z_{nk}的最大可能的上一状态$\varphi(z_{nk})$

$$\varphi(z_{nk}) = \arg\max_{j} \ln\left\{P\left(x_n|z_{n,k}\right)P\left(z_{n,k}|z_{n-1,j}\right)P\left(X_{n-1}, Z_{n-1,j}\right)\right\}$$

概率

$$P\left(X_n, Z_{n,k}\right) = P\left(x_n|z_{n,k}\right)P\left(z_{n,k}|z_{n-1,\varphi(z_{nk})}\right)P\left(X_{n-1}, Z_{n-1,\varphi(z_{nk})}\right)$$

$$P\left(X_1, Z_{1,k}\right) = P\left(x_1|z_{1,k}\right)\pi_k$$

当计算到序列尾端N时，z_N为最大可能序列：

$$z_N = \arg\max_{k} \ln\left\{P\left(X_N, Z_{N,k}\right)\right\}$$

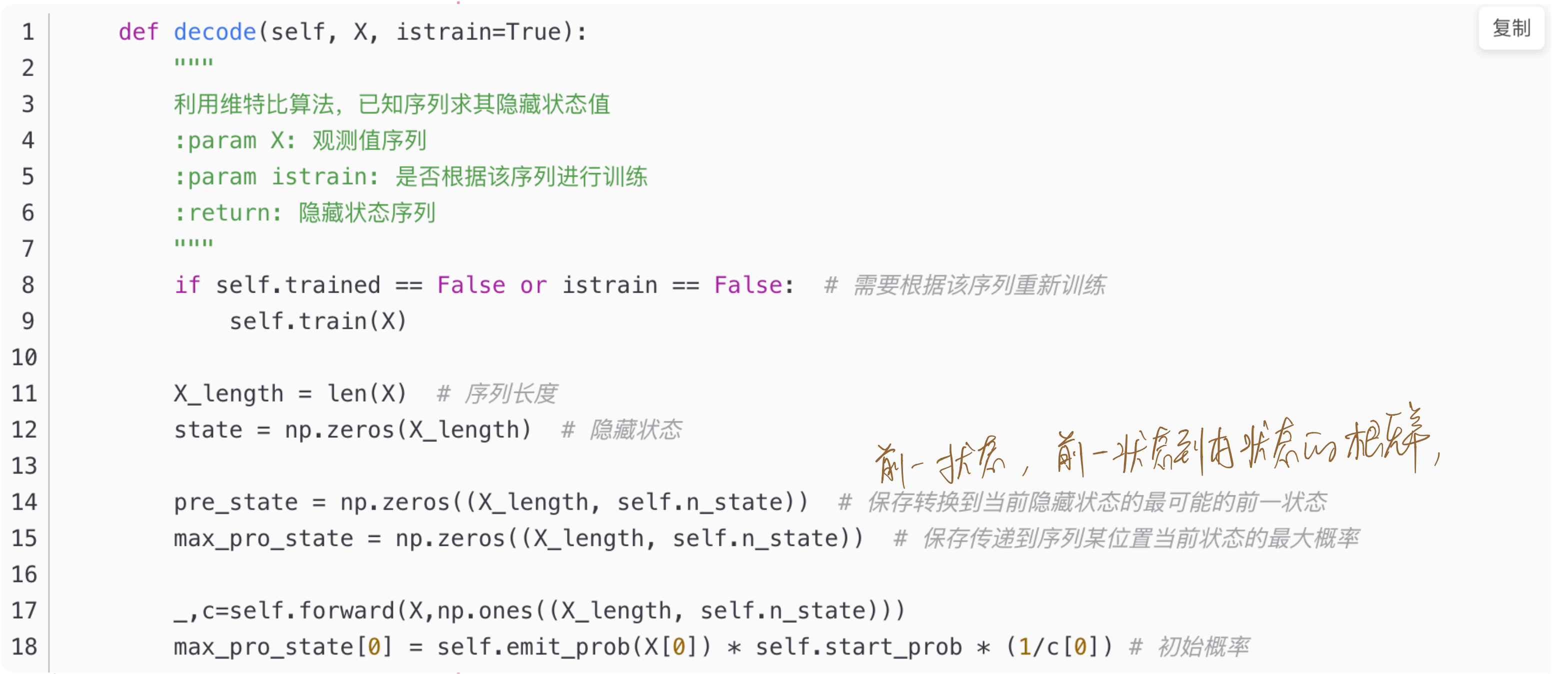
之后进行反向追踪法，序列n位置的隐藏状态为$\varphi(z_{n+1})$

注意到$P\left(X_{n-1}, Z_{n-1,j}\right)$会随着序列变长而逐渐迅速为 0，因此我们需要对其做归一化处理

$$\varphi(z_{nk}) = \arg\max_{j} \ln\left\{P\left(x_n|z_{n,k}\right)P\left(z_{n,k}|z_{n-1,j}\right)\frac{P\left(X_{n-1}, Z_{n-1,j}\right)}{P\left(X_{n-1}\right)}\right\}$$

$$\frac{P\left(X_n, Z_{n,k}\right)}{P\left(X_n\right)}\frac{P\left(X_n\right)}{P\left(X_{n-1}\right)} = \frac{P\left(X_n, Z_{n,k}\right)}{P\left(X_n\right)}c_n = P\left(x_n|z_{n,k}\right)P\left(z_{n,k}|z_{n-1,\varphi(z_{nk})}\right)\frac{P\left(X_{n-1}, Z_{n-1,\varphi(z_{nk})}\right)}{P\left(X_{n-1}\right)}$$

$$P\left(X_1, Z_{1,k}\right) = P\left(x_1|z_{1,k}\right)\pi_k$$



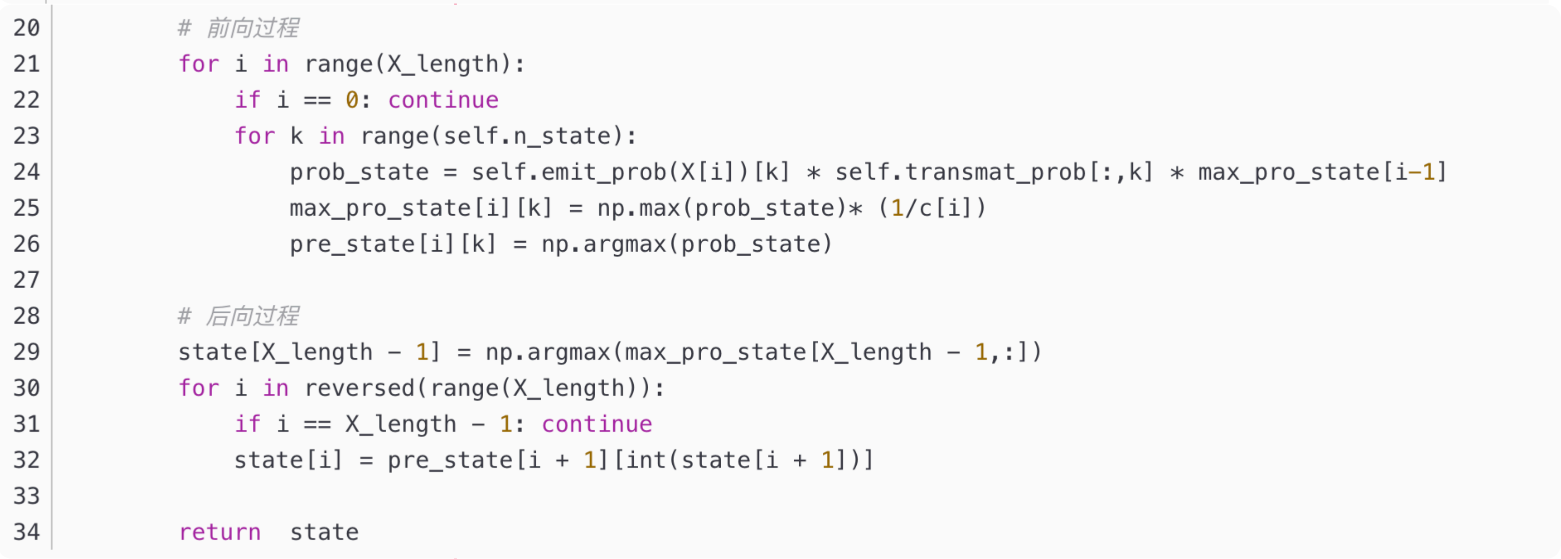
```python
    def decode(self, X, istrain=True):
        """
        利用维特比算法，已知序列求其隐藏状态值
        :param X: 观测值序列
        :param istrain: 是否根据该序列进行训练
        :return: 隐藏状态序列
        """
        if self.trained == False or istrain == False:  # 需要根据该序列重新训练
            self.train(X)

        X_length = len(X)  # 序列长度
        state = np.zeros(X_length)  # 隐藏状态

        pre_state = np.zeros((X_length, self.n_state))  # 保存转换到当前隐藏状态的最可能的前一状态
        max_pro_state = np.zeros((X_length, self.n_state))  # 保存传递到序列某位置当前状态的最大概率

        _,c=self.forward(X,np.ones((X_length, self.n_state)))
        max_pro_state[0] = self.emit_prob(X[0]) * self.start_prob * (1/c[0]) # 初始概率
```

前一状态，前一状态到由状态的概率，



```python
        # 前向过程
        for i in range(X_length):
            if i == 0: continue
            for k in range(self.n_state):
                prob_state = self.emit_prob(X[i])[k] * self.transmat_prob[:,k] * max_pro_state[i-1]
                max_pro_state[i][k] = np.max(prob_state)* (1/c[i])
                pre_state[i][k] = np.argmax(prob_state)

        # 后向过程
        state[X_length - 1] = np.argmax(max_pro_state[X_length - 1,:])
        for i in reversed(range(X_length)):
            if i == X_length - 1: continue
            state[i] = pre_state[i + 1][int(state[i + 1])]

        return  state
```