拉格朗日插值

拉格朗日插值

- 简介
- 公式
- 优化
  - ①关于x值连续时的优化
  - ②重心拉格朗日
  - ③快速插值
- 例题
  - luogu P4781【模板】拉格朗日插值
  - luoguP4593 [TJOI2018]教科书般的亵渎
  - 2019 ICPC 南昌邀请赛-Polynomial

参考资料

简介

在平面直角坐标系中，$n+1$个x坐标不同的点可以确认唯一的最高次为n的多项式。当我们要解决这些点求多项式的问题时，可以使用高斯消元来获得每一项的系数，但复杂度是$O(n^3)$的，而且往往会存在精度问题，而拉格朗日插值法可以在$O(n^2)$的复杂度内解决这一问题。

公式

假设多项式经过$(x_1,y_1),(x_2,y_2),(x_3,y_3)$三个点，那么可以假设

$f(x)=y_1\frac{(x-x_2)(x-x_3)}{(x_1-x_2)(x_1-x3)}+y_2\frac{(x-x_1)(x-x_3)}{(x_2-x_1)(x_2-x3)}+y_3\frac{(x-x_1)(x-x_2)}{(x_3-x_1)(x_1-x2)}$

这样我们直接把每个点代入，都只会剩下一项y，其正确性可以保证。

给出公式$f(k)=\sum_{i=0}^{n}\prod_{i\neq j}\frac{k-x[j]}{x[i]-x[j]}$，我们以此求出给定$k$时任一点的拟合$y$值。

优化

①关于x值连续时的优化

当x_i的取值是连续的时候，我们可以把式子优化成$O(n)$的复杂度。

把x_i换成$i\in[0,n]$，新的式子就是$f(k)=\sum_{i=0}^{n}y_i\prod_{i\neq j}\frac{k-j}{i-j}$

我们维护关于k的前缀积和后缀积$pre_i=\prod_{j=0}^{i}(k-j),suf_i=\prod_{j=i}^{n}(k-j)$

那么式子就变成了$f(k)=\sum_{i=0}^{n}y_i\frac{pre_{i-1}*suf_{i+1}}{fac[i]*fac[n-i]}$

不过这一优化适用条件太特殊了，不常用。

②重心拉格朗日

设$g_k=\prod_{i=1}^n k-x_i, t_i=\frac{y_i}{\prod_{j\neq i} x_i-x_j}$

这样式子就化成了$f(k)=g_k\sum_{i=0}^n\frac{t_i}{k-x_i}$

如此一来我们每次加入一个点时只需要记录t_i即可。

③快速插值

快速插值法可以在任意情况下做到$O(nlog^2n)$的时间复杂度的插值。

对于原公式$f(x)=\sum_{i=0}^n y_i\frac{\prod_{j\neq i}(x-x_j)}{\prod_{j\neq i}(x_i-x_j)}$，我们考虑如何快速求分子分母。

首先考虑分母：$g_i=\prod_{j\neq i}(x_i-x_j)=\lim_{x\to x_i}\frac{\prod_{j=0}^n(x-x_j)}{x-x_i}=(\prod_{j=0}^n(x-x_j)')|_{x=x_i}$

这样我们可以用分治求出$\prod_{j=0}^n(x-x_j)$，求导后对所有x_i多点求值即可。

求分子是同样的步骤，不过是把某项x_i换成特定的x值罢了。

例题

luogu P4781【模板】拉格朗日插值

给定n个点，确定一个$n-1$次多项式$y=f(x)$，并求$f(k)$

```
//#pragma GCC optimize("Ofast", "inline", "-ffast-math")
//#pragma GCC target("avx,sse2,sse3,sse4,mmx")
#include<bits/stdc++.h>
#define inf 0x3f3f3f3f
#define int long long
using namespace std;
const int N=2006;
const int mod=998244353;

int fpow(int a,int b){
    int res=1;
    while(b){
        if(b&1) res=res*a%mod;
        a=a*a%mod;
        b>>=1;
    }
    return res%mod;
}

int n,k,x[N],y[N];

int fc(int kk){
    int ans=0;
    for(int i=1;i<=n;i++){
        int fz=y[i],fm=1;
        for(int j=1;j<=n;j++){
            if(j==i) continue;
            fz=fz*(kk-x[j])%mod;
            fm=fm*(x[i]-x[j])%mod;
        }
        ans=(ans+fz*fpow(fm,mod-2)%mod+mod)%mod;
```

```
    }
    return ans%mod;
}

signed main(){
    scanf("%lld%lld",&n,&k);
    for(int i=1;i<=n;i++) scanf("%lld%lld",x+i,y+i);
    printf("%lld",fc(k));
    return 0;
}
```

luoguP4593 [TJOI2018]教科书般的亵渎

题意是求$\sum_{i=1}^{n} i^k$，可以看出这是一个关于n的$k+1$次多项式，可以插值解决。

把$n=0..k+1$的函数值求出来，就可以做到$O(k)$或者$O(klogk)$求n_0时的函数值。

其他求法：递推法、Bernoulli数、Stirling数、多项式差分

```
//#pragma GCC optimize("Ofast", "inline", "-ffast-math")
//#pragma GCC target("avx,sse2,sse3,sse4,mmx")
#include<bits/stdc++.h>
#define inf 0x3f3f3f3f
#define int long long
using namespace std;
const int N=66;
const int mod=1e9+7;

inline int read(){  int x=0,f=1;char ch=getchar();while(ch<'0'||ch>'9')
{if(ch=='-') f=f*-1;ch=getchar();}while(ch>='0'&&ch<='9'){x=x*10+ch-
'0';ch=getchar();}return x*f;}

int n,m,a[N],tot,inv[3601];


inline int fp(int a,int p){
    int res=1;
    while(p){
        if(p&1) res=res*a%mod;
        a=a*a%mod;
        p>>=1;
    }
    return res;
}

int x[N],y[N],fac[N],ifac[N],pre[N],suf[N];

inline int get(int nn,int mm){
    int lim=mm+1,ans=0;
    memset(y,0,sizeof(y));
    for(int i=1;i<=lim;i++) y[i]=(y[i]+y[i-1]+fp(i,mm))%mod;
    pre[0]=nn;suf[lim+1]=1;
    for(int i=1;i<=lim;i++) pre[i]=pre[i-1]*(nn-i)%mod;
    for(int i=lim;i>=1;i--) suf[i]=suf[i+1]*(nn-i)%mod;
    for(int i=0;i<=lim;i++){
        int up=pre[i-1]*suf[i+1]%mod*y[i]%mod,down=ifac[i]*ifac[lim-i]%mod;
```

```
        if((lim-i)&1) down=mod-down;
        ans=(ans+up*down%mod)%mod;
    }
    return ans;
}

inline void solve(){
    n=read();m=read();
    memset(a,0,sizeof(a));
    int ans=0;
    for(int i=1;i<=m;i++) a[i]=read();
    a[++m]=++n;
    sort(a+1,a+1+m);
    for(int i=1;i<=m;i++){
        for(int j=i;j<=m;j++) ans=(ans+get(a[j]-1,m)-get(a[j-1],m)+mod)%mod;
        for(int j=i+1;j<=m;j++) a[j]=(a[j]-a[i]+mod)%mod;
        a[i]=0;
    }
    printf("%lld\n",ans);
}

signed main(){
    inv[1]=1;for(int i=2;i<=3600;i++) inv[i]=(mod-mod/i)*inv[mod%i]%mod;
    fac[0]=1;for(int i=1;i<=60;i++) fac[i]=fac[i-1]*i%mod;
    ifac[60]=fp(fac[60],mod-2);
    for(int i=60;i>=1;i--) ifac[i-1]=ifac[i]*i%mod;
    int t=read();
    while(t--)  solve();
    return 0;
}
```

2019 ICPC 南昌邀请赛-Polynomial

已知$f(x)$是最高项为n次的多项式，给定$f(0)\dots f(n)$ 多次询问$\sum_{i=L}^{R} f(i) \mod 9999991$

连续插值的多项式优化得$f[k]=\sum_{i=0}^{n} y_i \prod_{i\neq j}\frac{k-j}{i-j}$,对于分母预处理阶乘和阶乘逆元，对于分子做前缀积和后缀积，化简得$f[k]=\sum_{i=0}^{n} y_i\frac{pre[i-1]*suf[i+1]}{fac[i]*fac[n-i]}[(n-i)\&1?-1:1]$

```
#include<bits/stdc++.h>
#define int long long
using namespace std;

const int N=1005;
const int MAXN=1e7+10;
const int mod=9999991;

int t,n,q,l,r;
int F[N],pre[N],suf[N],fac[N],ifac[N],sum[MAXN];

int fpow(int a, int b){
    int res=1;
    while(b){
        if(b&1) res=res*a%mod;
        a=a*a%mod;
```

```
        b>>=1;
    }
    return res;
}

void init(){
    fac[0]=1;
    for(int i=1;i<N;i++) fac[i]=fac[i-1]*i%mod;
    ifac[N-1]=fpow(fac[N-1],mod-2);
    for(int i=N-1;i>=1;i--) ifac[i-1]=ifac[i]*i%mod;
}

int cal(int *f, int k, int n){
    if(k<=n) return f[k];
    pre[0]=suf[n]=1;
    for(int i=1;i<=n;i++) pre[i]=pre[i-1]*(k-i+1)%mod;
    for(int i=n;i>=1;i--) suf[i-1]=suf[i]*(k-i)%mod;
    int ans=0;
    for(int i=0;i<=n;i++){
        int opt=(n-i)&1?-1:1;
        ans=(ans+opt*pre[i]%mod*suf[i]%mod*ifac[i]%mod*ifac[n-
i]%mod*f[i]%mod+mod)%mod;
    }
    return f[k]=ans;
}


signed main(){
    init();
    scanf("%lld",&t);
    while(t--){
        scanf("%lld%lld",&n,&q);
        for(int i=0;i<=n;i++) cin>>F[i];
        F[n+1]=cal(F,n+1,n);
        sum[0]=F[0];
        for(int i=1;i<=n+1;i++) sum[i]=(sum[i-1]+F[i])%mod;
        while(q--){
            scanf("%lld%lld",&l,&r);
            int ans=(cal(sum,r,n+1)-cal(sum,l-1,n+1)+mod)%mod;
            printf("%lld\n",ans);
        }
    }
    return 0;
}
```

参考资料

https://www.cnblogs.com/zwfymqz/p/10063039.html

https://blog.csdn.net/Code92007/article/details/94412729

https://www.cnblogs.com/ywwyww/p/8511505.html

https://www.cnblogs.com/dcdcbigbig/p/9715638.html

https://blog.csdn.net/fztsilly/article/details/109529465