简答题

降阶积分的定义？何时用降阶积分？优点有哪些？

以位移有限元为例。在单刚的计算中，需要我们在单元上进行积分操作。在积分过程中，希望在保证不损失收敛精度的前提下减少积分点个数，降低积分阶次。这样与精确积分（完全积分）相对比的积分手段被称为降阶积分（缩减积分）。

采用降阶积分能够改善模型过刚现象，并能够保持有限元精度，因此往往能够获得更好的解，因此广泛应用于各类问题，其中尤其是体积自锁及剪切自锁问题。但需要注意的是，在应用降阶积分时应当注意零能模态、K 奇异等现象的出现。

在保证不损失收敛精度的情况下，由于在每个积分点有大量计算的中型非线性模型， 可能值得考虑使用降阶积分。需要满足：精度由完全多项式；有限元本身是高估刚度矩阵。

特点：降阶积分可能会引起零能模态和 K 奇异。

不管一维、二维、三维，都采用 n=p-m+1 来确定积分阶次。

采用降阶积分往往可以得到更好的解，因为精确的积分是由非完全项决定的，而有限元的精度是由完全多项式决定的；有限元解偏刚，而降阶积分使模型刚度降低，有助于提高精度。

什么是零能模式？

在使用缩减积分时，位移模式并非刚体位移（即单元存在变形），但是其能量表达式$\int_V \frac{1}{2}\{\varepsilon\}^T[D]\{\varepsilon\}dV$的数值积分结果却等于 0（正常情况下应该大于 0）。

位移模式并非刚体位移，但是其能量表达式的数值积分结果却等于 0。（降阶积分可能会引起零能模式和 K 奇异）验证方法，对单个完全自由的单元求固有模态（固有振型），若存在非刚体运动的零特征值，则对应的模态就是零能模态，K 可能会奇异。

算法的定义？算法效率影响因素？算法设计要求？

定义：算法是用于求解严格确定的计算问题，能准确和全面理解的一些列指令，即给定初始状态或输入数据，经过计算机程序的有限次运算，能够得出所要求或期望的终止状态或输出数据。

效率影响因素：计算机本身的速度；依据的算法选用何种策略；问题的规模；程序语言；编译程序产生机器代码质量；机器执行指令速度。

算法设计要求：正确性（正确反应需求，通过测试）；可读性（有助于理解、调试和维护）；健壮性（完备的异常和出错处理）；高效率与低存储的需求（时间、空间的要求）。

Rtiz 法与有限元比较？

1、里兹法

是通过泛函驻值条件求未知函数的一种近似方法。

2、有限元法

有限元分析方法是使用有限元方法来分析静态或动态的物理物体或物理系统进行的分析方法。

二、方法不同

1、里兹法

是直接变分法的一种，以最小势能原理为理论基础。通过选择一个试函数来逼近问题的精确解，将试函数代入某个科学问题的泛函中，然后对泛函求驻值，以确定试函数中的待定参数，从而获得问题的近似解。

2、有限元法

有限元分析是用较简单的问题代替复杂问题后再求解。它将求解域看成是由许多称为有限元的小的互连子域组成，对每一单元假定一个合适的（较简单的）近似解，然后推导求解这个域总的满足条件（如结构的平衡条件），从而得到问题的解。

三、应用不同

1、里兹法

这一方法在许多力学、物理学、量子化学问题中得到应用。在机械工程领域，它被用于计算多自由度系统（如弹簧-质量系统、变截面轴上的飞轮）大致的共振频率；还可以计算圆柱体的折断载荷。

2、有限元法

有限元法在工程设计和科研领域得到了越来越广泛的重视和应用，已经成为解决复杂工程分析计算问题的有效途径，从汽车到航天飞机几乎所有的设计制造都已离不开有限元分析计算，其在机械制造、材料加工、航空航天、汽车、土木建筑、电子电器、国防军工、船舶、铁道、石化、能源和科学研究等各个领域的广泛使用已使设计水平发生了质的飞跃。

1. 简述里兹法与有限元比较

里兹法即在某一函数空间寻找试探函数，利用加权值的方法使泛函的驻值问题转化为权值的极值问题。

里兹法在全域选取试探函数，解精度取决于试探函数选取。有限元法是单元级别的里兹法，试探函数在分片连续单元上选取，仅需在单元上满足边界条件。

① 相同：本质相同，均为泛函变分，利用极值求待定参数。

② 不同：A 里兹法近似函数在全域定义，只能求解简单边界；有限元法近似函数在子域定义，适用于复杂边界。

B 里兹法试探函数项数越多精度越高；有限元法可通过提高近似函数阶次，通过缩小单元尺寸提高精度。

C 里兹法一般要求写出试探函数表达式；有限元法只要求得各节点上未知函数值。

里兹法和有限元法：　　(泛函的极小值对应于给定边界上的控制方程).

1. 性质不同，

里兹法是通过泛函驻值条件求未知函数的一种近似方法

有限元法是使用有限元方法来分析静态或动态的物理物体或物理系统进行的分析方法。

2. 方法不同.

里兹法是变分法的一种，以最小势能原理为理论基础，通过选择一个试函数来逼近问题的精确解，将试函数代入某个科学问题的泛函中，然后对泛函求驻值，以确定试函数中的待定参数，从而获得问题的近似解.

有限元法是用比较简单的问题代替复杂问题然后求解。它将求解域看成是由许多称为有限元的小的互连子域组成，对每个单元假定一个适合的近似解，然后推导求解这个域总的满足条件，从而得到问题的解.

有限元法和经典里兹法的不同之处在试探函数的定义上，在经典里兹法和伽辽金法中，试探函数由定义在全域上的一组基函数组成，这种组合必须能够表示真实解，也必须满足边界条件。在有限元法中，试探函数是由定义在组成全域的子域上的一组基函数组成。因为子域很小，所以定义在子域上的基函数能够十分简单。

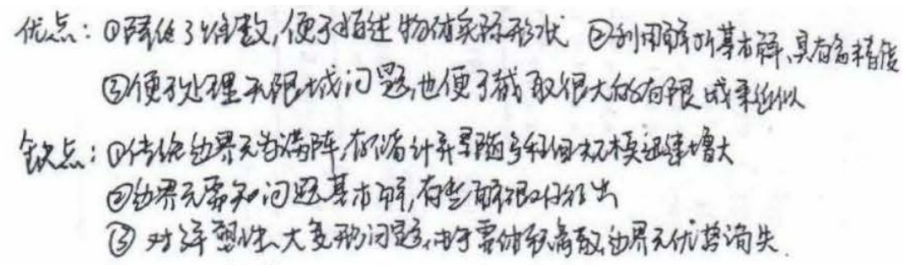
与有限元法相比，边界元的优缺点。

优点：降低了维数，便于描述物质实际，对边界物理量求解精度有极高要求的问题有压倒性优势；利用解析基本解，具有高精度；便于处理无限域问题，也便于截取很大的有限域集近似。

缺点：常规边界元法不能大规模求解，其计算量随规模迅速增大；边界元需要

知道问题的基本解，有些解很难给出；对弹塑性大变形问题，由于需求体积离散边界元优势丧失。

边界元法的优点是：将全解域的计算化为解域边界上的计算，使求解问题的维数降低了一维，减少了计算工作量；能够方便地处理无界区域问题。例如对于势流等的无限区域问题，使用边界元法求解时由于基本解满足无穷远处边界条件，在无穷远处边界上的积分恒等于零。因此对于无限区域问题，例如具有无穷远边界的势流问题，无需确定外边界，只需在内边界上进行离散即可；边界元法的精度一般高于有限元法。边界元法的主要缺点是边界元方程组的系数矩阵是不对称的满阵，该方法目前只适用于线性问题。



相比有限元，边界元的优缺点

优点：有限元属于区域法，其剖分涉及到整个区域，而边界元只需对边界离散，因此，可以降低求解问题的维数；有限元法待求未知数多，要求解的方程规模大，导致输入数据多，计算的准备工作量大，边界元法则相对规模小一些；有限元必须同时对所有域内节点和边界节点联立求解，边界元只需对边界节点联立求解，然后可以相互独立、完全并行的计算域内各点的函数值；

缺点：有限元的系数矩阵带状稀疏，且保持对称正定性，边界元法的矩阵为满矩阵，一般不能保证正定对称性；有限元适应复杂的几何形状和边界条件，适于求解非线性、非匀质问题，边界元仅适应规则区域及边界条件，适于求解线性、匀质问题；有限元适合于求解有界区域无奇异性问题，而边界元适合于求无界区域问题及若干奇异性问题；对于狭长区域，有限元的精度高于边界元，其它情况下，边界元的精度较高。

有限元法与伽辽金法的比较？

（1）有限元基于 Ritz 方法。首先将连续体划分成离散单元，并在单元内部建立插值基函数，用于描述单元内部的函数变化。因此对于有限元法与伽辽金法的比较，实质上是 Ritz方法与伽辽金方法的比较。

（2）Ritz方法基于等效泛函理论。在应用过程中，一般希望寻找原方程所对应的二次泛函，并将线性问题转化为二次泛函求极值问题，在单元内部建立含有参数的插值函数，最后通过取变分的方法来计算待定参数，最后确定分片插值函数的形式。

（3） 伽辽金方法基于加权余量法。加权余量法在建立过程中希望能够使残差与权函数内积的积分形式为零值。不同的权函数的取法一般对应不同的名称，对于伽辽金法，仅要求积分域内权函数与试函数一致，边界则取反值。随后即可构建求解方程。

写出原始伽辽金的形函数和等效积分形式，如何构造对称格式？

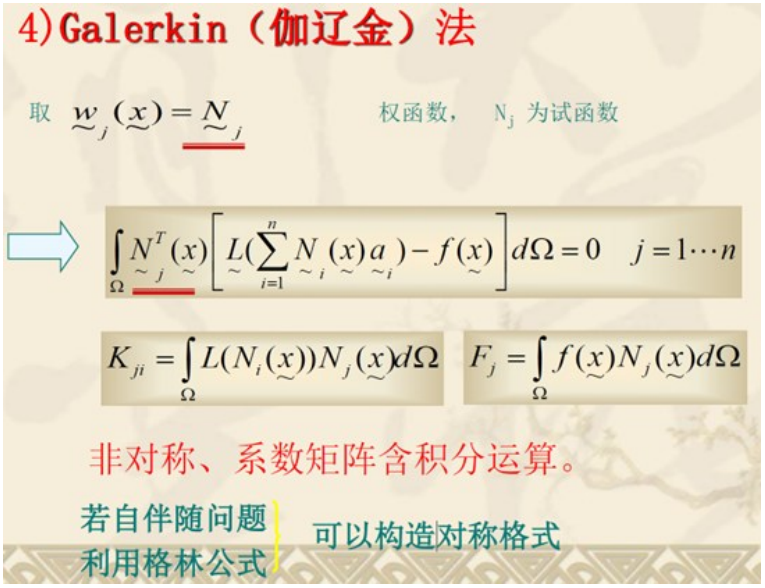
第三部分 1，P19；若自伴随问题利用格林公式可以构造对称格式；

伽辽金法：

取 $\{w_j\}=\{N_j\}$

$\{\bar{w}_j\}=-\{N_j\}$

$$\int_\Omega \{N_j\}^T \{A(\tilde{N}\tilde{a})\} d\Omega - \int_C \{N_j\}^T \{B(\tilde{N}\tilde{a})\} ds = 0 \qquad (j=1,2,\ldots,n)$$



(5) 伽辽金法

取 $\boldsymbol{W}_j=\boldsymbol{N}_j$，在边界上 $\overline{\boldsymbol{W}}_j=-\boldsymbol{W}_j=-\boldsymbol{N}_j$。即简单地利用近似解的试探函数序列作为权函数。近似积分形式(1.2.19)式可写成

$$\int_\Omega \boldsymbol{N}_j^{\mathrm{T}}\mathbf{A}\left(\sum_{i=1}^{n}\boldsymbol{N}_i a_i\right)\mathrm{d}\Omega-\int_\Gamma \boldsymbol{N}_j^{\mathrm{T}}\mathbf{B}\left(\sum_{i=1}^{n}\boldsymbol{N}_i a_i\right)\mathrm{d}\Gamma=\mathbf{0}\quad(j=1,2,\cdots,n)\qquad(1.2.22)$$

由(1.2.16)式，可以定义近似解 $\tilde{\boldsymbol{u}}$ 的变分 $\delta\tilde{\boldsymbol{u}}$ 为

$$\delta\tilde{\boldsymbol{u}}=\boldsymbol{N}_1\delta a_1+\boldsymbol{N}_2\delta a_2+\cdots+\boldsymbol{N}_n\delta a_n\qquad(1.2.23)$$

其中 δa_j 是完全任意的。由此(1.2.22)式可更简洁地表示为

$$\int_\Omega \delta\tilde{\boldsymbol{u}}^{\mathrm{T}}\mathbf{A}(\tilde{\boldsymbol{u}})\mathrm{d}\Omega-\int_\Gamma \delta\tilde{\boldsymbol{u}}^{\mathrm{T}}\mathbf{B}(\tilde{\boldsymbol{u}})\mathrm{d}\Gamma=\mathbf{0}\qquad(1.2.24)$$

对于近似积分的“弱”形式(1.2.21)式则有

$$\int_\Omega \mathbf{C}^{\mathrm{T}}(\delta\tilde{\boldsymbol{u}})\mathbf{D}(\tilde{\boldsymbol{u}})\mathrm{d}\Omega+\int_\Gamma \mathbf{E}^{\mathrm{T}}(\delta\tilde{\boldsymbol{u}})\mathbf{F}(\tilde{\boldsymbol{u}})\mathrm{d}\Gamma=\mathbf{0}\qquad(1.2.25)$$

将会看到，如果算子 $\mathbf{A}$ 是 $2m$ 阶的线性自伴随的(见 1.3.1 节)，采用伽辽金法得到的求解方程的系数矩阵是对称的，这是在用加权余量法建立有限元格式时几乎毫无例外地采用伽辽金法的主要原因，而且当微分方程存在相应的泛函时，伽辽金法与变分法往往导致同样的结果。

伽辽金法所建立的求解方程是否一定对称？简要说明。

从一般性而言，尽管伽辽金方法对于权函数的选择作出了特殊要求，但并不能保证求解方程一定对称。不过对于自伴随问题而言，由于我们总是能够通过分部积分（或格林公式）的方法来处理算子的作业，因此这类问题最后往往可以转化为对称问题。

什么是多项式算法，与指数算法比较？

“多项式”、“指数”都是用来描述问题复杂度的指标。

多项式算法：设有解某种问题的一个算法，对于输入长度为 L 的具体问题，其

计算步骤有一个上界 P(L)，若 P(y)是 y 的多项式，则称此算法是一个多项式时间算法，简称多项式算法。若为 y 的指数函数，则为指数时间算法。

需要注意的是当问题复杂度随输入规模的增加呈多项式地增长时，这个算法才被认为是有效的。

指数算法：如果一个算法的运行时间或空间复杂度随着输入规模 n 的增加呈指数增长，那么该算法称为指数算法。具体地，复杂度可以表示为 $O(2^n)$或 $O(c^n)$，其中 c 是一个常数，且 c> 1。

当输入规模增大时，任意一个多项式算法终将变得比指数算法更有效。

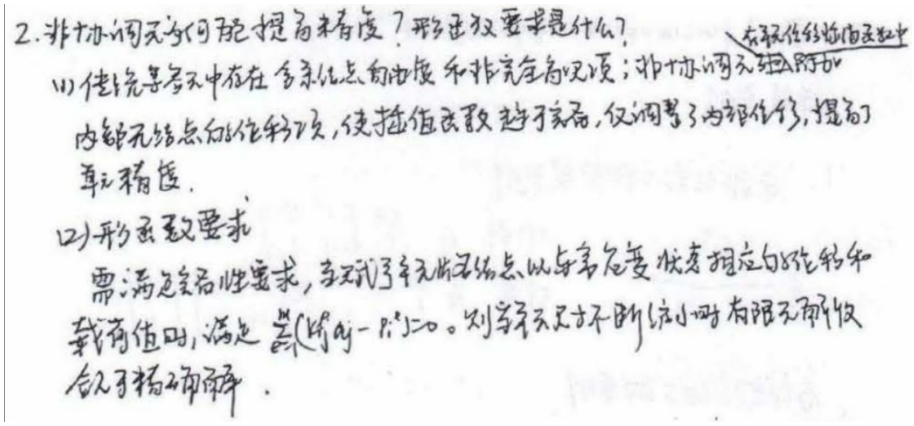
非协调元为什么能提高精度？形函数的要求是什么？

（此种单元把增强单元位移梯度的附加自由度引入线性单元，能克服线性完全积分中的剪切自锁问题，具有较高的计算精度。）非协调元：为了改善二维线性单元的性质，提高其精度，Wilson 提出在单元的位移插值函数中附加内部无结点的位移项，使得在单元与单元的交界面上是不保证协调的，也就是说由于单元内增加了附加位移项而致使单元之间不能保证在交界面上位移的连续性。这些附加位移项称之为非协调项，引入非协调位移项的单元称为非协调元。

其特点如下：

1. 非协调元的位移插值函数中附加内部无结点的位移项；
2. 非协调元不满足有限元中的相容性要求，在单元与单元的交界面上是不保证协调的；
3. 通过分片实验的非协调元可以保证解的收敛；
4. 可以改善单元性质，提高精度。

形函数要求：二次项完全



最小势能原理、最小余能原理，其应力、应变、位移满足什么条件？

最小势能原理：位移和应变满足小变形几何条件；最小余能原理：应力满足平

衡方程及边界条件。

最小势能原理要求位移为可能位移，即具有适定光滑性且满足域内的位移协调方程以及位移边界条件的位移场被称为可能位移。对于应变场，一般要求位移场至少具有 C1 连续性，因此应变场一般要求有 C0 连续性，即应变函数本身的连续。

线性方程解法几大类？代表性解法有哪些？

两大类：直接法与迭代法。

直接法（消去法；分解法）：Gauss 消去法、 约当消去法、三对角方程组追赶法、迭代改善；LL^T 分解法、LDL^T 分解法。

迭代法：Jacobi 迭代法、Gauss-Siedel 迭代法、松弛因子迭代法、共轭斜量法。对称高斯赛德尔（SGS）、krylov 子空间法（广义最小残差法 GMRES、共扼梯度法 CG）。

有约束条件的泛函有什么方法，有什么特点？

拉格朗日乘子法、罚函数法。

拉格朗日乘子法：是驻值问题；大多数情况下有物理意义

罚函数法：原泛函有极小值；不进行变分；对 α 取值有要求，不同的取值会带来一些问题，α 越大，约束条件满足的越好。

区别和比较。处理约束的方式：罚函数法：通过在目标函数中加入一个罚项来处理约束。约束被违反时，目标函数的值会显著增加，从而惩罚不满足约束的解。拉格朗日乘数法：通过引入拉格朗日乘数直接将约束纳入目标函数中，形成拉格朗日函数。求解时同时考虑目标函数和约束条件。

方法性质：罚函数法：将约束优化问题转化为无约束优化问题，适合于各种优化方法。随着罚因子变大，解会趋于满足约束条件。拉格朗日乘数法：直接处理约束优化问题，拉格朗日乘数提供了有关约束活跃性的额外信息。适合于解

析解法和一些特定的数值方法。

解的性质：罚函数法：由于罚因子需要趋向于无穷大，可能会导致数值不稳定性和病态问题。需要选择合适的罚因子以确保计算的稳定性。拉格朗日乘数法：提供更直接的解法，特别是在解析方法中。数值上更稳定，但求解的非线性系统可能更复杂。

应用场景：罚函数法：适用于不易直接处理约束的复杂优化问题，以及需要将约束问题转化为无约束问题的情况。拉格朗日乘数法：适用于解析解法、简单的线性或非线性约束问题，以及需要直接处理约束条件的优化问题。

动力响应的解法有哪些？

直接积分法（中心差分法、Newmark 方法、Wilson-θ 法）；振型叠加法。

大型矩阵存储格式有哪些，总刚度矩阵特点？

压缩行存储、压缩列存储、压缩块存储、对称储存格式、二维等带宽存储或一维变带宽存储、行主元稀疏存储、细胞稀疏存储。

特点：稀疏性，总刚度矩阵的绝大部分元素为零，因为每个有限元只与其相邻的单元和节点产生相互作用；对称性：对于许多物理问题，总刚度矩阵是对称的。这是因为作用力与反作用力的关系；带状结构：总刚度矩阵通常具有带状结构，即非零元素集中在对角线附近。这是因为每个节点通常只与其直接相邻的节点相互作用；正定性：总刚度矩阵通常是正定的，特别是在结构力学和热传导等问题中。这意味着其所有特征值都是正的；分块结构：在多物理场耦合问题中，总刚度矩阵可以分为多个子块，每个子块对应一个物理场或一个子系统。

简述三角形、四面体网格生成过程。

三角形网格算法：P110；四面体网格算法：P118。

四边形 / 六面体　　　三角形 / 四面体

映射　　　剖分　分网

- Octree　八叉树
- Advancing Front　前沿推进
- Delaunay　三角化.

① 平面四叉树：

作框 → 分一半、细分、再分 → 有洞、再细分 → 连接格点 → 去掉全部三角形在洞内部分.

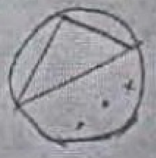
② 前沿推进：

1. 定义边界上节点（可按要求）
2. 从边界向内推进（选择尺寸）　　全部

↓

连接起来，作出ABC，扔掉　再新边界

③ Delaunay：　空圆 / 最大特性　：外接圆不含其他节点



步骤：1. 假设有初始网格，再新增加一个点

2. 将点与三角形顶点连接.

往边向外看，组成四边形.

3. 对角交换.

法2：定位.

检查周围节点分布，构成多边形.

将内部网格删掉，

分别和其他几点相连

a)Lawson 算法

逐点插入的 Lawson 算法 [2] 是 Lawson 在 1977 年提出的，该算法思路简单，易于编程实现。基本原理为：首先建立一个大的三角形或多边形，把所有数据点包围起来，向其中插入一点，该点与包含它的三角形三个顶点相连，形成三个新的三角形，然后逐个对它们进行空外接圆检测，同时用 Lawson 设计的局部优化过程 LOP 进行优化，即通过交换对角线的方法来保证所形成的三角网为 Delaunay 三角网。

上述基于散点的构网算法理论严密、唯一性好，网格满足空圆特性，较为理想。由其逐点插入的构网过程可知，遇到非 Delaunay 边时，通过删除调整，可以构造形成新的 Delaunay 边。在完成构网后，增加新点时，无需对所有的点进行重新构网，只需对新点的影响三角形范围进行局部联网，且局部联网的方法简单易行。同样，点的删除、移动也可快速动态地进行。但在实际应用当中，这种构网算法当点集较大时构网速度也较慢，如果点集范围是非凸区域或者存在内环，则会产生非法三角形。

如左图 4 所示：当离散点集构成圆环时，Lawson 算法产生的非法三角形

b)Bowyer-Watson 算法(推荐)

Watson 算法的基本步骤是：

1、构造一个超级三角形，包含所有散点，放入三角形链表。

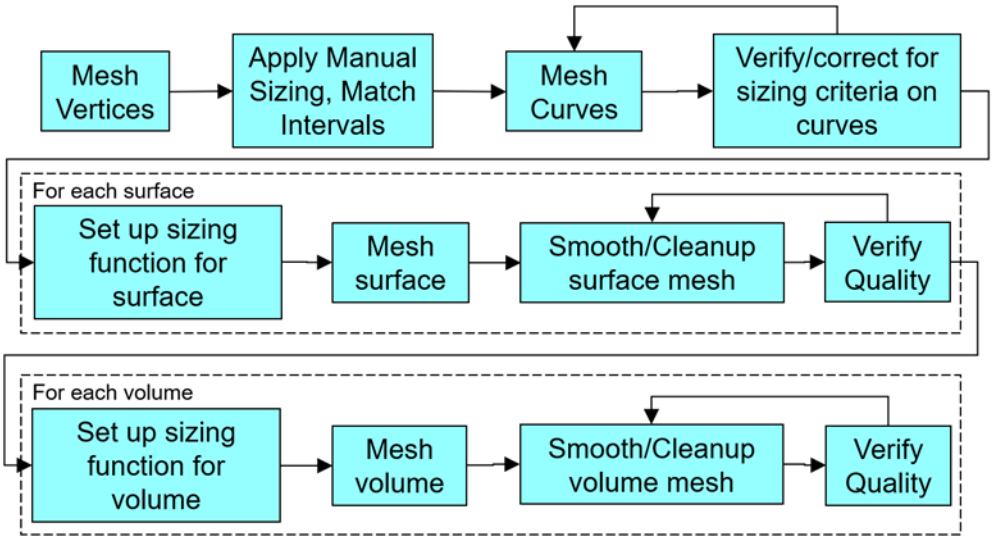
2、将点集中的散点依次插入，在三角形链表中找出外接圆包含插入点的三角形（称为该点的影响三角形），删除影响三角形的公共边，将插入点同影响三角形的全部顶点连接起来，完成一个点在 Delaunay 三角形链表中的插入。

3、根据优化准则对局部新形成的三角形优化。将形成的三角形放入 Delaunay 三角形链表。

4、循环执行上述第 2 步，直到所有散点插入完毕。

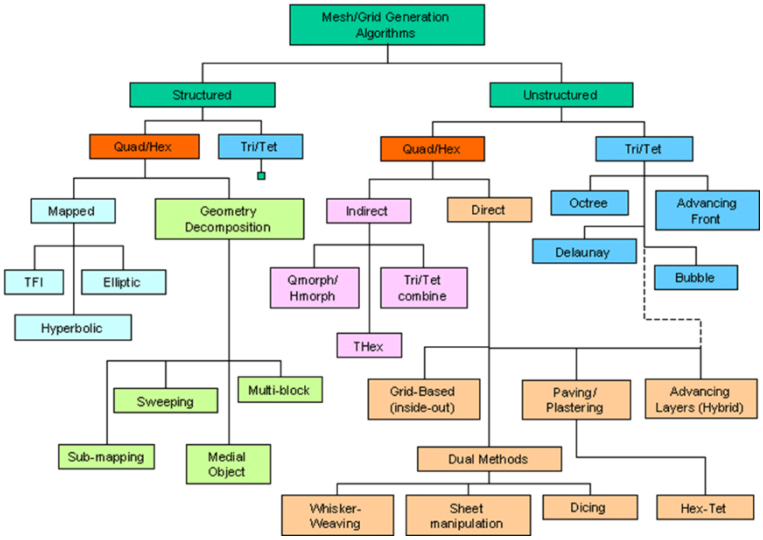
这一算法的关键的第 2 步图示如下：

3. Mesh Generation Process



The Mesh Generation Process

18



体网格根据其单元形状可以分为四面体网格（tetra-mesh），六面体网格（hexa-mesh），以及四面体或六面体为主的多面体网格（tetra/hexa-dominated mesh）。而根据其生成方式又可以分为结构化与非结构化网格（stuctured, non-structured），贴体与非贴体网格（conformal, non-conformal），等等。

对于四面体网格，较为常用的生成算法包括Delaunay法和波前法（advancing front）等。而对于六面体网格，较为常用的算法有：映射法（mapping），扫掠法（sweeping），以及正交切割单元法（Cartesian cut-cell）。映射法和扫掠法只适用于特定类型的几何模型；而正交切割单元法具有较强的普适性，只需要提供模型的表面网格就可以自动生成六面体为主的多边形网格（并非纯六面体网格）

有些体网格生成算法直接从描述几何模型的参数方程出发（例如映射法，扫掠法），而另一类体网格生成算法从模型的表面网格出发（例如Delaunay法，波前法），正交切割单元法属于后者。它对于输入的面网格有一些要求：
1）纯三角（triangular）：所有单元均为三角形。
2）满足水密性条件（water-tight）：任意一条边恰好属于两个三角形单元。
3）满足流形条件（manifold）：任意两个三角形要么不相交，要么共享一条边。
算法最终得到的是一组网格单元（cut-cell element），每个网格单元由一个正方体和若干个切割面（cut-face）共同构成。在必要时可以计算出正方体被切割后形成的多面体，所有这些多面体共同构成模型内部空间区域的一个剖分。

算法的大致流程如下：
1）初始化：首先根据面网格的包围盒（bounding box），以及用户指定的分辨率（resolution）生成均匀的正方体单元，作为第一层网格单元的基础。
2）切割：对于输入的面网格中的每个三角形，确定所有与其相交的正方体单元，并计算出被正方体切割后得到的切割面（cut-face），储存在相应的正方体单元中。每个三角形上的所有切割面构成这个三角形的一个剖分。
3）细分：对于在模型内部而未与表面网格相交的正方体单元，可直接放入体网格当中。对于与表面网格相交的正方体单元，根据其存储的切割面数量和法向，决定是否将其细分（refine）。对于满足细分条件的单元，将其分成大小相等的8个小正方体单元，并将原本单元中存储的所有切割面再次切分后，分别存储到对应的小单元之中。
4）组装：当所有正方体单元均不满足细分条件，或者达到预设的最大细分次数时，细分过程终止。将所有含有切割面的单元放入体网格当中。此时，体网格中包含了所有完全位于模型内部的单元，以及与模型表面相交的单元，它们构成模型内部空间区域的一个剖分。

网格节点→应用手册大小，匹配间隔→网格曲线→修正曲线上的尺寸标准→（对于每个面）→对面建立尺寸函数→划分表面网格→光滑表面网格→验证质量→（对于每个体）→对体建立尺寸函数→划分体网格→光滑体网格→验证质量

网格优化方法（拓扑优化、几何优化）

【课件：网格剖分】

拓扑优化：拓扑优化主要通过调整网格的连接关系和元素形状，改变网格的拓扑结构来优化网格。

边交换（Edge Swapping）边交换是一种常用的拓扑优化方法，主要用于三角形和四面体网格。

局部重划分（Local Remeshing）局部重划分通过在局部区域重新生成网格来优化网格。方法：选择局部区域，移除该区域的网格元素，然后重新生成网格。

多面体重构（Polyhedral Reconstruction）多面体重构是一种高级的拓扑优化方法，主要用于复杂几何结构的网格优化。方法：将现有网格元素重构为多面体，并根据几何特征重新划分网格。

几何优化：几何优化主要通过调整网格点的位置，改变网格的几何形状来优化网格。

Laplacian 平滑（Laplacian Smoothing）Laplacian 平滑是一种简单而有效的几何优化方法，通过移动网格点到其相邻点的平均位置来平滑网格。

形状优化（Shape Optimization）形状优化通过调整网格点的位置，优化网格形状，以提高网格的质量和数值稳定性。

平衡优化（Equilibrium Smoothing）平衡优化通过模拟物理平衡过程，调整网格点的位置，使网格达到最佳状态。

综合优化方法：在实际应用中，通常将拓扑优化和几何优化结合使用，以获得更好的优化效果。常用的综合优化方法包括：

迭代优化（Iterative Optimization）迭代优化方法通过交替进行拓扑优化和几何优化，逐步提高网格质量。

自适应网格优化（Adaptive Mesh Optimization）自适应网格优化方法根据物理场的分布，动态调整网格密度和质量。

广义特征值问题的求解在有限元中的应用？

在有限元中，包括动力响应中的固有振动分析、振动响应分析以及稳定性屈曲分析等问题均属于广义特征值问题。

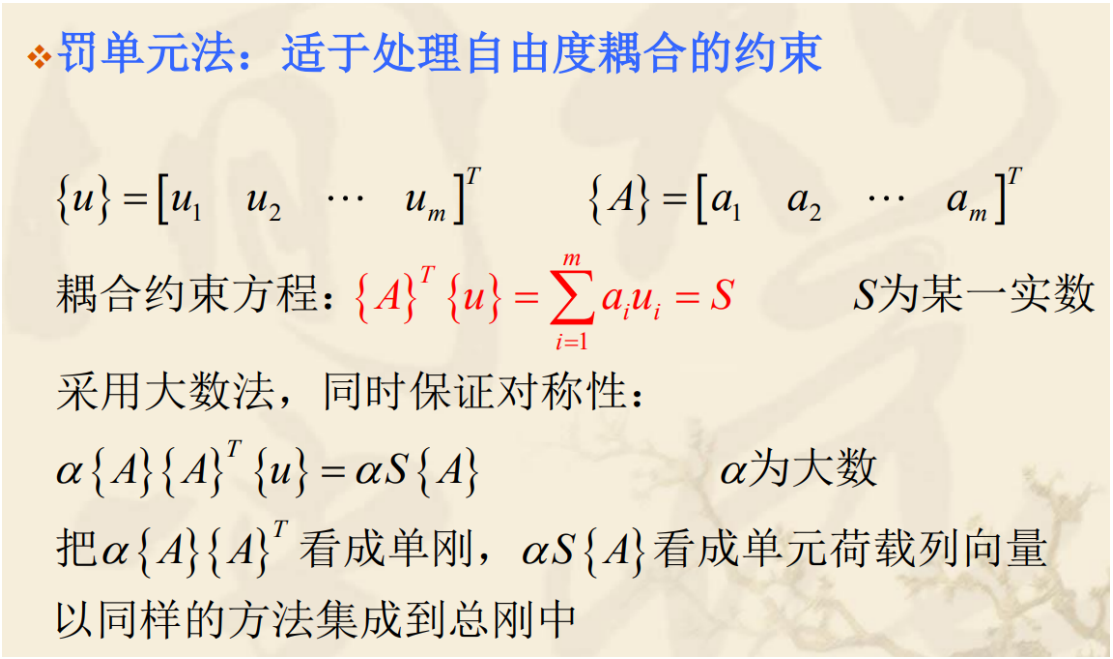
等参元和次参元的定义，是否满足收敛准则？

（1） 在单元坐标变换的过程中，存在包括确定几何形状的节点（m 个）以及确定场函数的节点（n 个）。若 m=n，则为等参元；m>n，超参元；m<n，亚（次）参元。

（2） 收敛准则要求单元内部的插值函数能够满足完备性和协调性。其中协调性的满足只需要网格划分合理即可保证。而对于完备性要求，需要验证对于 C0 单元，在母单元中若要求场函数包含一次多项式，在变换后的子单元中能否满足，即能够重现线性场。经过验证可知，对于等参元与亚参元总是可以满足的，因此在这样的条件下往往能够满足收敛准则。

若某些节点自由度存在线性约束，如何处理？

线性约束对应物理模型中的弹簧单元，属于自由度耦合的约束。参考约束优化问题的罚函数方法，在处理时一般引入惩罚单元：



1. 拉格朗日乘数法

拉格朗日乘数法是一种通过引入拉格朗日乘数来处理约束的数值方法。

步骤：

1. **引入拉格朗日乘数**：
   假设线性约束条件为 $C\mathbf{u} = d$，其中 C 是约束矩阵，$\mathbf{u}$ 是节点位移向量，d 是常数向量。

2. **构建拉格朗日函数**：
   引入拉格朗日乘数向量 λ，构建新的拉格朗日函数：

   $$L(\mathbf{u}, \lambda) = \frac{1}{2}\mathbf{u}^T K\mathbf{u} - \mathbf{u}^T\mathbf{f} + \lambda^T(C\mathbf{u} - d)$$

   其中，K 是刚度矩阵，$\mathbf{f}$ 是载荷向量。

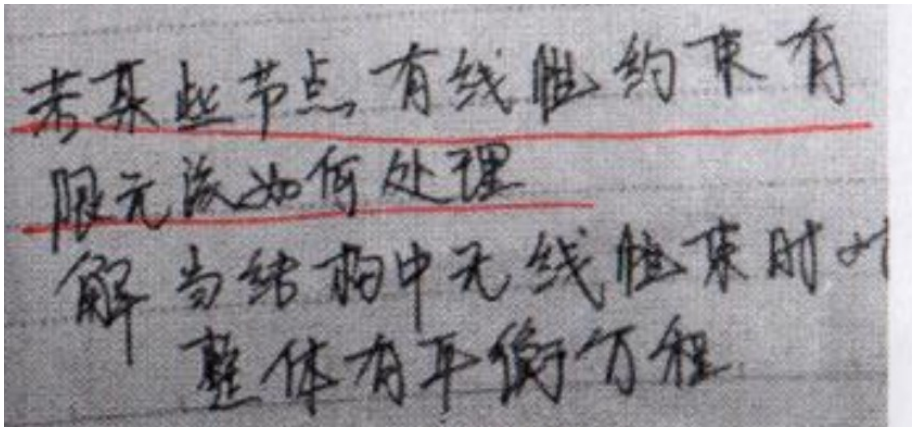
3. **求解拉格朗日函数的驻点**：
   对 $\mathbf{u}$ 和 λ 求偏导数并令其为零，得到以下方程组：

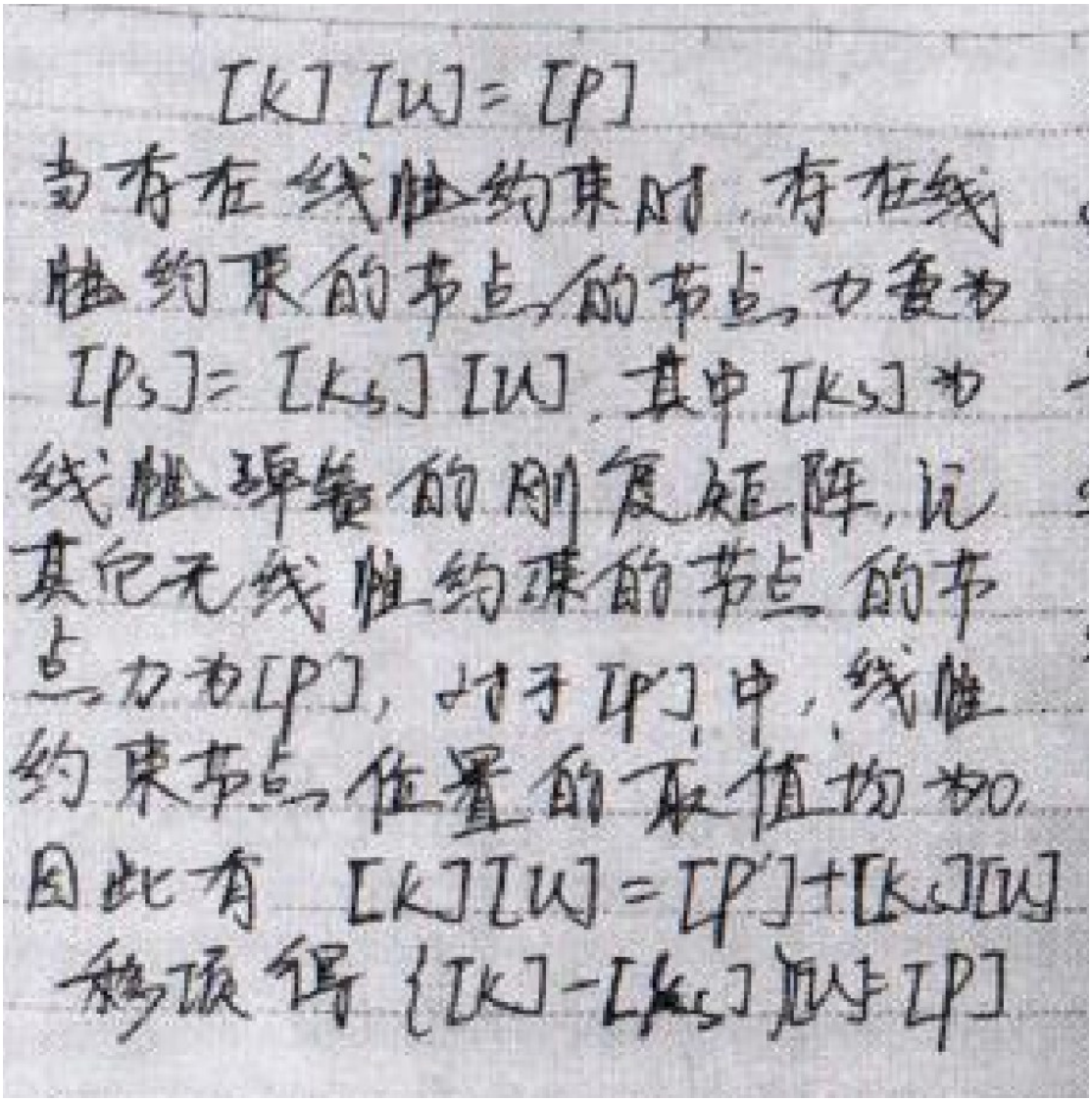
   $$\begin{cases} K\mathbf{u} + C^T\lambda = \mathbf{f} \\ C\mathbf{u} = d \end{cases}$$

4. **组合方程组**：
   将上述方程组组合成一个增广矩阵形式：

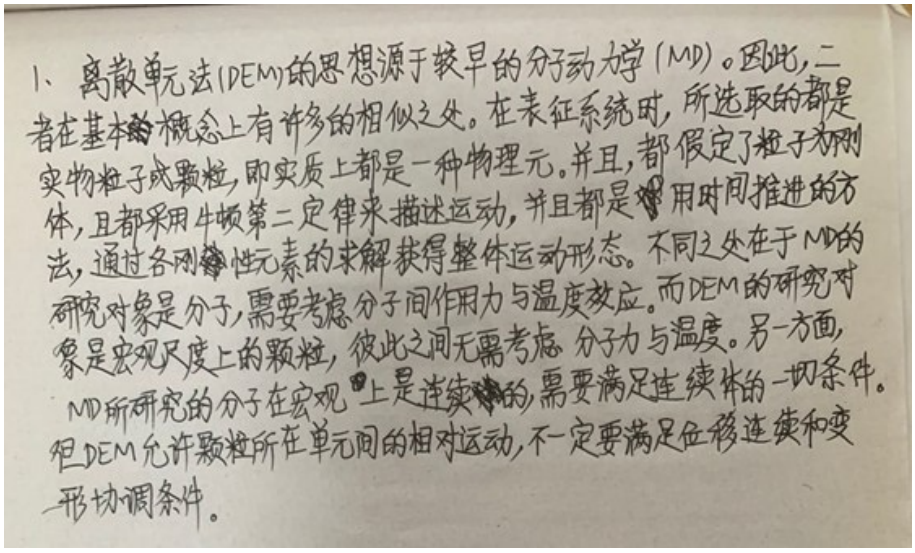
   $$\begin{bmatrix} K & C^T \\ C & 0 \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ d \end{bmatrix}$$

5. **求解增广方程组**：
   使用数值方法求解该增广方程组，得到位移向量 $\mathbf{u}$ 和拉格朗日乘数向量 λ。

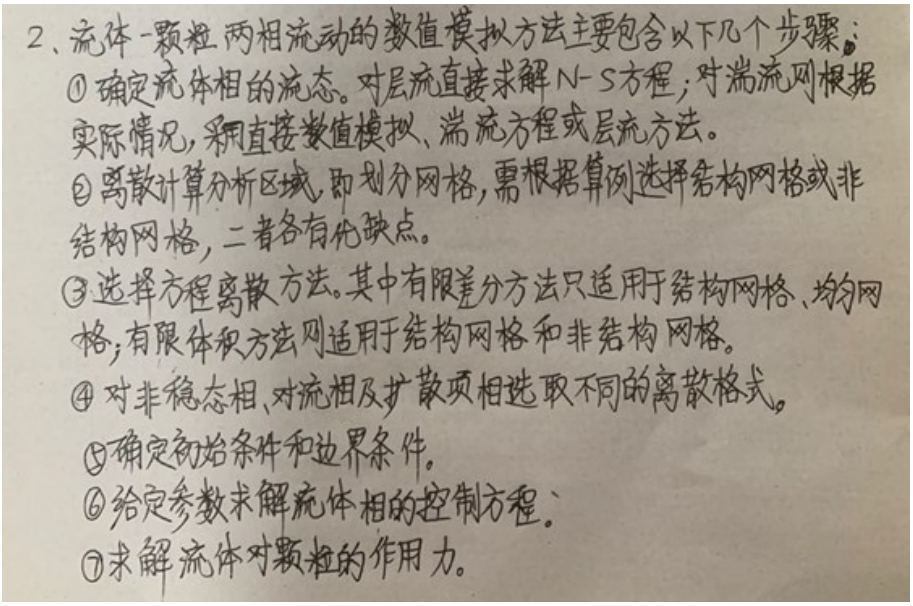




离散单元法 DEM 和分子动力学 MD 的相似和不同之处？


1、离散单元法(DEM)的思想源于较早的分子动力学(MD)。因此，二者在基本的概念上有许多的相似之处。在表征系统时，所选取的都是实物粒子成颗粒，即实质上都是一种物理元。并且，都假定了粒子为刚体，且都采用牛顿第二定律来描述运动，并且都是用时间推进的方法，通过各刚性元素的求解获得整体运动形态。不同之处在于MD的研究对象是分子，需要考虑分子间作用力与温度效应。而DEM的研究对象是宏观尺度上的颗粒，彼此之间无需考虑分子力与温度。另一方面，MD所研究的分子在宏观上是连续的，需要满足连续体的一切条件。但DEM允许颗粒所在单元间的相对运动，不一定要满足位移连续和变形协调条件。

颗粒流两相流动算法？



SPH 方法模拟时的精度影响因素？

当使用 SPH（光滑粒子动力法）模拟时，影响因素包括：

（1）粒子分布。粒子分布的不均匀性会影响模拟结果的精度，粒子集中分布的区域模拟结果一般更准确。

（2）内部或边界颗粒。如果内部或边界颗粒与周围粒子存在强烈的相互作用，则可能导致局部区域的粒子密度异常增加或减少，从而引起预测结果的偏差。

（3）核函数类型的选择。核函数的选择要求能够满足一系列包括光滑性、紧支性的数学性质，否则会对模拟的准确性和稳定性具有较大影响。

（4）光滑长度的选择。光滑长度是核函数的重要参数之一，决定了核函数的形状以及每个粒子与周围粒子的相互作用范围。过小的光滑长度会导致只有距离该粒子非常近的粒子会对其产生相互作用，这样可能导致局部粒子密度过高，从而引起数值震荡和不稳定性。过大的光滑长度会导致每个粒子会与过多的周围粒子产生相互作用，这样可能导致局部粒子密度过低，从而影响模拟结果的准确性。

（5）运用 SPH 方法时求解域不能被边界截断。

①离散格式。高阶的。

②粒子的分布。只要不是特别的无序分布，修正矩阵，如果高度无序，则修正矩阵可能变态。

③和网格类方法类似，与网格的分布相关，非规则的网格，畸形的粒子的分布可以规则化，规则的粒子变得规则化。

粒子类方法更适用并行。

DEM 求解过程？

接触检查；颗粒作用力的计算；求解颗粒运动速度及位移；进入下一个时步。

Veriet 算法推导，优缺点？

Verlet算法推导

Verlet算法基于牛顿运动方程：

$$\mathbf{F} = m\mathbf{a}$$

其中，$\mathbf{F}$ 是力，m 是质量，$\mathbf{a}$ 是加速度。

考虑物体在时间 t 处于位置 $\mathbf{r}(t)$，其在时间 $t + \Delta t$ 和 $t - \Delta t$ 处的位置分别为 $\mathbf{r}(t + \Delta t)$ 和 $\mathbf{r}(t - \Delta t)$。利用泰勒展开公式，我们可以得到以下两个表达式：

$$\mathbf{r}(t + \Delta t) = \mathbf{r}(t) + \mathbf{v}(t)\Delta t + \frac{1}{2}\mathbf{a}(t)\Delta t^2 + O(\Delta t^3)$$

$$\mathbf{r}(t - \Delta t) = \mathbf{r}(t) - \mathbf{v}(t)\Delta t + \frac{1}{2}\mathbf{a}(t)\Delta t^2 - O(\Delta t^3)$$

将这两个方程相加，可以消去速度项：

$$\mathbf{r}(t + \Delta t) + \mathbf{r}(t - \Delta t) = 2\mathbf{r}(t) + \mathbf{a}(t)\Delta t^2$$

重排并解得：

$$\mathbf{r}(t + \Delta t) = 2\mathbf{r}(t) - \mathbf{r}(t - \Delta t) + \mathbf{a}(t)\Delta t^2$$

在每个时间步长 Δt 处，根据当前位置 $\mathbf{r}(t)$、前一位置 $\mathbf{r}(t - \Delta t)$ 和当前加速度 $\mathbf{a}(t)$ 来计算下一位置 $\mathbf{r}(t + \Delta t)$。

Verlet算法的优缺点

优点

1. **时间可逆性**：Verlet算法是时间对称的，即时间反演对称性（time-reversible），这意味着如果我们从 $\mathbf{r}(t + \Delta t)$ 开始，能够准确地回到 $\mathbf{r}(t)$ 和 $\mathbf{r}(t - \Delta t)$。这种特性使得它在长期模拟中表现出色，保持能量和动量的稳定。

2. **高稳定性**：Verlet算法在长时间模拟中保持能量守恒，避免了能量漂移问题。这对需要精确计算总能量的系统（如分子动力学）特别重要。

3. **简单实现**：算法简单，只需要存储当前位置和前一位置，适合大规模计算。

4. **无速度依赖性**：在基本形式中，Verlet算法不显式计算速度，这减少了存储需求和计算复杂度。

缺点

1. **初始条件敏感性**：由于算法依赖于初始位置和前一位置，因此需要提供两个初始位置或一个初始位置和初始速度。这可能会增加实现复杂度。

2. **精度较低**：尽管Verlet算法具有良好的能量守恒性，但其局部截断误差为二阶（$O(\Delta t^2)$），整体误差随时间积累。因此在高精度要求下，可能需要较小的时间步长。

3. **速度计算复杂**：基本形式不显式计算速度。如果需要速度，必须通过后处理步骤计算，这可能增加额外计算开销。

Verlet 与 Velocity，Verlet 算法的格式与优缺点？

- **Verlet algorithm**

$$r(t+h) = -r(t-h) + 2r(t) + h^2F(t)/m + O(h^4)$$

■ 优点：

1、精确，误差$O(\Delta^4)$

2、每次积分只计算一次力

3、时间可逆（ time-reversible ）

→most commonly used!

■ 缺点：

1、速度有较大误差$O(\Delta^2)$

2、轨迹与速度无关，无法与温度（热浴）耦联

- **Velocity Verlet algorithm**

$$r(t+h) = r(t) + h\,v(t) + h^2\,F(t)/2\,m + O(h^3)$$
$$v(t+h) = v(t) + h\,(F(t)/m + F(t+h)/m)/2 + O(h^3)$$

- Explicitly includes velocities 明确包括速度信息
- Self-starting from the positions and velocities at the initial time. 从初始时刻的位置和速度自启动
- Mathematically identical to the original Verlet algorithm. 在数学上与原始的Verlet算法完全相同
- Do not need to store the values of r(t) and v(t) at two different times. 不需要在两个不同的时间存储r(t)和v(t)的值

Frequently used in real MD simulations !

- Need to calculate new forces F(t+h) after calculation of new positions r(t+h) but before calculation of new velocities v(r+h).

需要在计算新的位置r(t+h)之后，但在计算新的速度v(t+h)之前计算新的力F(t+h)。

SPH 光滑长度自适应，推到粒子类方程

WCSPH 与 ISPH 的优缺点？

（刘 PPT 流体动力学）

ISPH：压力看起来很精确，计算量很大

WCSPH：压力看起来不精确，计算量很小

WCSPH 有良好的近似方法，良好的边界处理和压力光滑技术

颗粒材料的特点及如何描述颗粒运动？

（离散元）

颗粒流体两相流动算法？

（作业题）

粒子粗粒化过程？

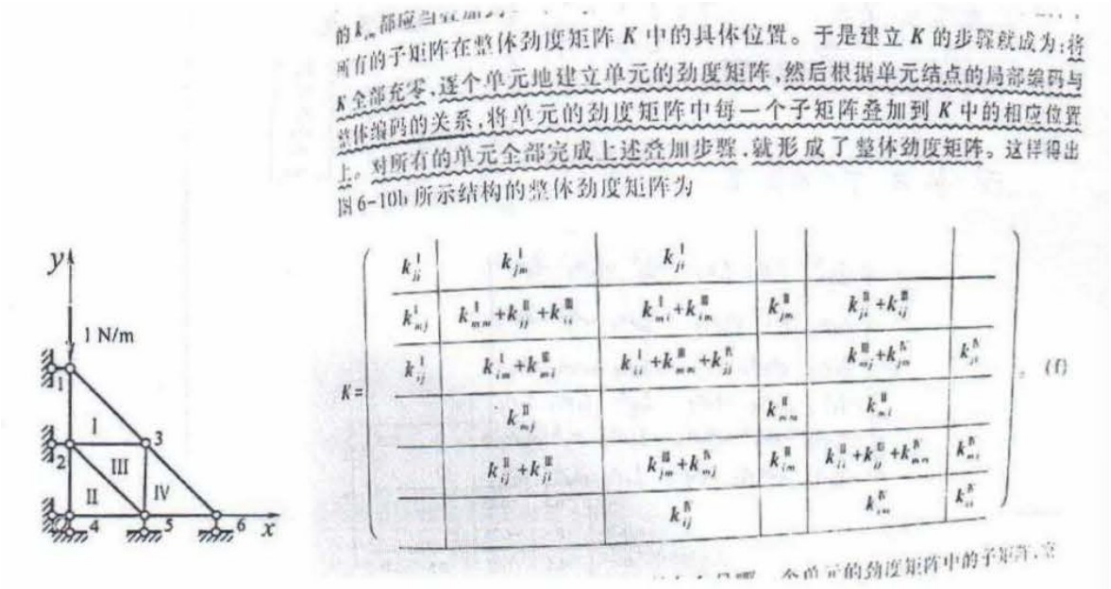
（作业题/11 周粗粒耗散粒子动力学 P8）

粒子模拟指利用计算机模拟分子或宏观物质的运动以及相互作用。粗粒化基于计算机图形学，将粒子一些琐碎的分子、原子级别的细节忽略掉，保留主要的物理特征，将一群分子、原子视作较大的粒子，降低计算的复杂度。

在粗粒化过程中，需要将原子模型映射到粗粒化模型，并粗粒化粒子之间的相互作用，通过粗粒化模型再现目标函数，并优化粗粒化模型中的参数、函数，最终进行粗粒化模拟。通过粗粒化在保持模拟结果准确性的同时，能够降低计算成本，是一种常用的模拟手段。

①确定目标，确定粗粒化程度②将原子模型映射到粗粒度模型③粗粒之间的相互作用④通过粗粒度模型再现目标函数⑤优化粗粒度模型中的参数、函数⑥进行粗粒模拟。

整体刚度矩阵如何整合？



的 k_{im} 部应当在[illegible]

所有的子矩阵在整体劲度矩阵 K 中的具体位置。于是建立 K 的步骤就成为：将 K 全部充零，逐个单元地建立单元的劲度矩阵，然后根据单元结点的局部编码与整体编码的关系，将单元的劲度矩阵中每一个子矩阵叠加到 K 中的相应位置上。对所有的单元全部完成上述叠加步骤，就形成了整体劲度矩阵。这样得出图 6-10b 所示结构的整体劲度矩阵为

$$K=\begin{pmatrix} k_{ii}^{\mathrm{I}} & k_{im}^{\mathrm{I}} & k_{ji}^{\mathrm{I}} & & & \\ k_{mi}^{\mathrm{I}} & k_{mm}^{\mathrm{I}}+k_{ij}^{\mathrm{II}}+k_{ii}^{\mathrm{III}} & k_{mi}^{\mathrm{I}}+k_{im}^{\mathrm{III}} & k_{jm}^{\mathrm{III}} & k_{ji}^{\mathrm{III}}+k_{ij}^{\mathrm{III}} & \\ k_{ij}^{\mathrm{I}} & k_{im}^{\mathrm{I}}+k_{mi}^{\mathrm{III}} & k_{ii}^{\mathrm{I}}+k_{mm}^{\mathrm{III}}+k_{jj}^{\mathrm{IV}} & & k_{mj}^{\mathrm{III}}+k_{jm}^{\mathrm{IV}} & k_{ji}^{\mathrm{IV}} \\ & k_{mj}^{\mathrm{II}} & & k_{mm}^{\mathrm{II}} & k_{mi}^{\mathrm{II}} & \\ & k_{ij}^{\mathrm{II}}+k_{ji}^{\mathrm{III}} & k_{jm}^{\mathrm{III}}+k_{mj}^{\mathrm{IV}} & k_{im}^{\mathrm{III}} & k_{ii}^{\mathrm{II}}+k_{jj}^{\mathrm{III}}+k_{mm}^{\mathrm{IV}} & k_{mi}^{\mathrm{IV}} \\ & & k_{ij}^{\mathrm{IV}} & & k_{im}^{\mathrm{IV}} & k_{ii}^{\mathrm{IV}} \end{pmatrix} \quad (f)$$

各单元的劲度矩阵中的子矩阵，它