Coordinate Systems

Coordinate systems, operators, and transformations.

Kevin Cowtan
cowtan@ysbl.york.ac.uk

Coordinate systems

- We have to deal with many coordinate systems in crystallographic software. The book-keeping is not exciting, but it is vital. Simplifications (e.g. assuming orthogonal grids) must usually be paid for later.
- Examples:
  - Orthogonal Ångstrom coordinates.
  - Fractional coordinates.
  - Reciprocal orthogonal coordinates.
  - Reflection (Miller) indices, i.e. *HKL*s.
  - Grid coordinates.

Kevin Cowtan, cowtan@ysbl.york.ac.uk Sienna/Coordinate Systems

Coordinate Systems

- In addition, we need to handle:
  - Transformations between these coordinate systems.
  - Transformations within a coordinate system (i.e. rotation and translation operators).
  - Rotation representations.
  - Derivatives of functions with respect to different coordinate systems.
- This lecture will give a basic overview of the issues. Implementations of all these data types and transformations are a part of both the **CCTBX** and **Clipper** crystallographic libraries.

Kevin Cowtan, cowtan@ysbl.york.ac.uk Sienna/Coordinate Systems

Coordinate Systems

- Conventions for this lecture:
  - scalars are italic, lower case, e.g. *s, r*
  - vectors are italic, lower case and underlined, e.g. $\underline{v}$, $\underline{x}$, $\underline{h}$
  - matrices are italic, uppercase and bold, e.g. $\boldsymbol{O}$, $\boldsymbol{F}$, $\boldsymbol{M}$
- In addition to matrix notation, most equations are also given as explicit sums of terms.
  - The elements of a vector or matrix are given by the same symbol in italic lower case with an appropriate number of subscripts. e.g. x_i, $O_{i,j}$.
- All vectors and matrices are of rank 3.

Kevin Cowtan, cowtan@ysbl.york.ac.uk Sienna/Coordinate Systems

Coordinate Systems

Coordinate systems:

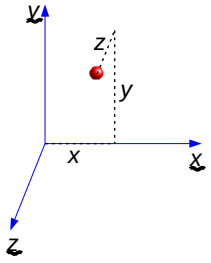
- Coordinate are used to describe the positions of elements within the crystal system. e.g. the position of a particular atom within a unit cell, or a particular grid point within a grid.
- 3 dimensions -> rank 3.
  - Coordinates represented by a vector of 3 numbers.

Kevin Cowtan, cowtan@ysbl.york.ac.uk Sienna/Coordinate Systems

Coordinate Systems: Real space

Orthogonal Ångstrom Coordinates:

- 3 orthogonal distances in Ångstroms along directions $\underline{x}$ $\underline{y}$ $\underline{z}$ (Formally: basis vectors)



$$\underline{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

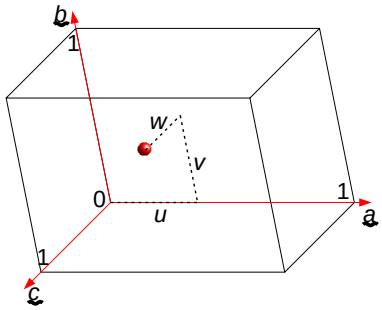
Note: there is no reference to the unit cell at this point.

Kevin Cowtan, cowtan@ysbl.york.ac.uk Sienna/Coordinate Systems

Coordinate Systems: Real space

Fractional coordinates:

- Position in the unit cell described as a fractional position along each cell edge:



$$\underline{u} = \begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}$$

Note: since the cell repeats, u,v,w repeat on the range 0...1. We often standardize on the range 0...1 (or -1/2...1/2), but this may split a molecule.

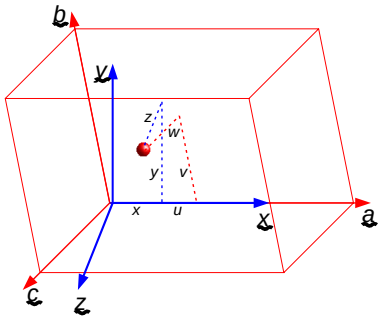
Coordinate Systems: Real space

Relating orthogonal and fractional coordinates:

- We can orient and position one coordinate system however we want with respect to the other, *but...*
- It is convenient to adopt some convenient convention.
- The most common convention in the PDB (also the CCP4 default) is:
  - Align the $\underline{a}$ axis along $\underline{x}$
  - Align the $\underline{b}$ axis in the $\underline{x}$-$\underline{y}$ plane
    - Or equivalently align $\underline{z}$ axis perpendicular to $\underline{a}$ and $\underline{b}$

Coordinate Systems: Real space

Relating orthogonal and fractional coordinates:



Coordinate Systems: Real space

Relating orthogonal and fractional coordinates:

- An orthogonal coordinate may be determined from a fractional coordinate by:

$$\underline{x} = \boldsymbol{O}\,\underline{u}$$

i.e.

$$x_i = \Sigma_j O_{ij} u_j$$

$\underline{x}$ = $\boldsymbol{O}$ $\underline{u}$

Where $\boldsymbol{O}$ is the orthogonalization matrix. For the common convention,

$$\boldsymbol{O} = \begin{pmatrix} a & b\cos(\gamma) & c\cos(\beta) \\ 0 & b\sin(\gamma) & -c\sin(\beta)\cos(\alpha^*) \\ 0 & 0 & c\sin(\beta)\sin(\alpha^*) \end{pmatrix}$$

Coordinate Systems: Real space

Relating orthogonal and fractional coordinates:

- A fractional coordinate may be determined from an orthogonal coordinate by:

$$\underline{u} = \boldsymbol{F}\,\underline{x}$$

i.e.

$$u_i = \Sigma_j F_{ij} x_j$$

$\underline{u}$ = $\boldsymbol{F}$ $\underline{x}$

Where $\boldsymbol{F}$ is the fractionalization matrix.
Clearly $\boldsymbol{F} = \boldsymbol{O}^{-1}$

- Note:
  - $a,b,c,\alpha,\beta,\gamma$ are the cell constants.
  - $a^*,b^*,c^*,\alpha^*,\beta^*,\gamma^*$ are the reciprocal cell constants.

Coordinate Systems: Real space

Measuring distances:

- We do this all the time, e.g. inter-atomic distances.
- In orthogonal coordinates, the squared distance between two points is given by $r^2 = \Delta x^2+\Delta y^2+\Delta z^2$

i.e: $\underline{\Delta x} = \underline{x}_2 - \underline{x}_1$

$$r^2 = \underline{\Delta x}^T \underline{\Delta x}$$

or:

$$r^2 = \Sigma_i \Delta x_i^2$$

r^2 = $\underline{x}^T$ $\underline{x}$

Coordinate Systems: Real space

Measuring distances:

- For the distance between two fractional coordinates, convert to orthogonal first:

$$r^2 = \Delta\underline{u}^T \boldsymbol{O}^T \boldsymbol{O} \Delta\underline{u}$$

r^2	=	$\underline{u}^T$	$\boldsymbol{O}^T$	$\boldsymbol{O}$	$\underline{u}$

or:

$$r^2 = \Sigma_i \Sigma_j \Sigma_k \, O_{ij} \, O_{ik} \, \Delta u_j \, \Delta u_k$$

- Simplify by pre-calculating the central product:

$$\boldsymbol{M} = \boldsymbol{O}^T \boldsymbol{O}$$

$$M_{jk} = \Sigma_i \, O_{ij} \, O_{ik}$$

- $\boldsymbol{M}$ is a symmetric matrix, called the "real-space metric tensor".

Coordinate Systems: Real space

Measuring distances:

- Simplified form using the metric tensor:

$$r^2 = M_{11} \Delta u_1^2 + M_{22} \Delta u_2^2 + M_{33} \Delta u_3^2 + 2 M_{12} \Delta u_1 \Delta u_2 + 2 M_{13} \Delta u_1 \Delta u_3 + 2 M_{23} \Delta u_2 \Delta u_3$$

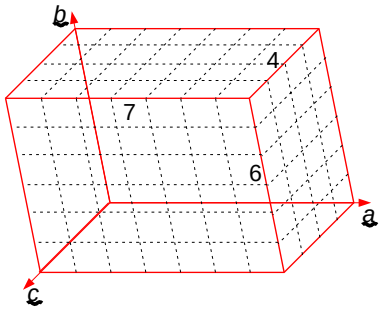
(since the matrix is symmetric, we just use the upper triangle and double the off-diagonal terms).

- (This is often a performance critical task).

Coordinate Systems: Real space

Other coordinate types:

- Grid coordinates: Electron density maps are usually calculated on a grid which samples the unit cell



$$\underline{n} = \begin{pmatrix} n_u \\ n_v \\ n_w \end{pmatrix} = \begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix}$$

$$\text{e.g.} \quad \underline{n} = \begin{pmatrix} 7 \\ 6 \\ 4 \end{pmatrix}$$

Coordinate Systems: Real space

Other coordinate types:

- Grid coordinates: $\underline{g}$
  - Each grid point is indexed using 3 integer indices, g_i, usually starting from 0.
  - The sampling usually involves a grid in which each cell edge is divided into a set number of equal divisions, with the number of divisions roughly proportional to the cell edge.
  - Symmetry and FFT requirements may constrain these values (e.g. multiple of 2, 3, 4, no large prime factors).

Coordinate Systems: Real space

Other coordinate types:

- Grid coordinates:
  - Convert to grid coordinates by scaling the fractional coordinates by the samplings, and taking nearest integer:

$$g_i = \text{int}(n_i u_i)$$

$$u_i = g_i / n_i$$

  - For orthogonal coordinates , convert to/from fractional first. As an optimization, the two steps can be combined.
  - Grid coordinates repeat every n_i along the i'th axis.

Coordinate Systems: Real space

Other coordinate types:

- Grid coordinates: Additional complications:
  - M. Rowicka, A. Kudlicki and Z. Otwinowski, [Acta Cryst. (2002). A58, 574-579] use grids which do not intersect the origin to improve symmetry handling in the FFT
  - Hexagonal close packed grids give a more efficient sampling of real space. How are they best indexed?

Coordinate Systems: Real space

Other coordinate types:

- Map coordinates: (Cowtan)
  - Non-integer grid coordinates.
    - For crystallographic maps, fractional coordinates do the job just fine
    - For non-crystallographic maps, fractional coordinates are undefined.
  - Used for interpolation in non-crystallographic maps.

Coordinate Systems: Real space

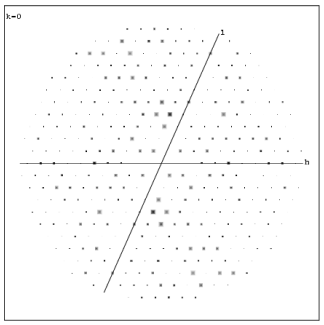
Implementation issues:

- It is very easy when programming to make mistakes over coordinate types.
- When using strongly typed languages (e.g. C++), implement each coordinate type as a different class to prevent such errors.
  - (Use inheritance for common behaviors)
- Coordinate types or cell and sampling classes may then implement all the required conversions.

See CCTBX or Clipper.

Coordinate Systems: Reciprocal space

In reciprocal space we mainly deal with reflections, indexed by integer *h,k,l*:



$$\underline{h} = \begin{pmatrix} h \\ k \\ l \end{pmatrix} = \begin{pmatrix} h_1 \\ h_2 \\ h_3 \end{pmatrix}$$

Coordinate Systems: Reciprocal space

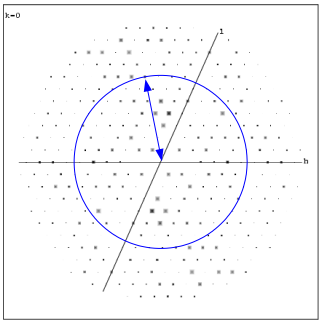
In reciprocal space we mainly deal with reflections, indexed by integer *h,k,l*:

- *h,k,l* are coordinates on a non-orthogonal grid, like grid coordinates.
- *h,k,l* do not repeat. They are centered (+/-) about the origin *h=k=l=0*.
- May be referred to as Miller indices (by correspondence with the indexing of crystal cell faces).

Coordinate Systems: Reciprocal space

We frequently need to determine the (reciprocal) distance from a reflection to the origin (resolution).

We may also need to determine a reciprocal orthogonal coordinate e.g. to make this picture.



Coordinate Systems: Reciprocal space

Relating reciprocal orthogonal and fractional coordinates:

- An reciprocal orthogonal coordinate may be determined from an HKL by:

$$\underline{s} = \boldsymbol{O}^* \underline{h}$$

- As in real space, $\boldsymbol{O}^*$ is determined by the orthogonalization convention. A convenient choice is to use the transpose of the real space fractionalizing matrix: $\boldsymbol{O}^* = \boldsymbol{F}^T$
  - i.e. $\underline{z}$ parallel to $\underline{c}^*$, $\underline{y}$ in the $\underline{b}^*$-$\underline{c}^*$ plane

Coordinate Systems: Reciprocal space

Relating reciprocal orthogonal and fractional coordinates:

- For display purposes, the "Cambridge Convention" is more common:
  - $\underline{x}$ parallel to $\underline{a}^*$
  - $\underline{y}$ in the $\underline{a}^*$-$\underline{b}^*$ plane
- In this case $\boldsymbol{O}^*$ is calculated using the equivalent formula to $\boldsymbol{O}$.
- **Don't mix conventions between real and reciprocal space – it will only end in tears.**

Coordinate Systems: Reciprocal space

Measuring distances in reciprocal space:

- As before, to calculate squared distances (in inverse squared Ångstroms), we first need reciprocal orthogonal coordinates, or a reciprocal metric tensor:

$$s^2 = \underline{h}^T \boldsymbol{O}^{*T} \boldsymbol{O}^* \underline{h}$$

or:

$$s^2 = \Sigma_i \Sigma_j \Sigma_k O^*_{ij} O^*_{ik} h_j h_k$$

- Simplify by pre-calculating the central product:

$$\boldsymbol{M}^* = \boldsymbol{O}^{*T} \boldsymbol{O}^*$$

$$M^*_{jk} = \Sigma_i O^*_{ij} O^*_{ik}$$

Coordinate Systems: Reciprocal space

Measuring distances:

- Simplified form using the metric tensor:

$$s^2 = M^*_{11} h^2 + M^*_{22} k^2 + M^*_{33} l^2 + 2 M^*_{12} h k + 2 M^*_{13} h l + 2 M^*_{23} k l$$

- Resolution in Ångstroms is $1/s$, i.e. $\sqrt{1/s^2}$
- $s^2 = 4 \sin^2\theta/\lambda^2$
- Some developers use the symbol s instead of s^2.
- Clipper and CCP4 refer to s^2 as inverse resolution squared: "invresolsq".

Interlude

The Structure Factor Equation:

- From Scattering Theory:

$$\underline{F}(\underline{s}) = \Sigma_j f_j(\underline{s}) \exp(2\pi i \, \underline{s}^T \underline{x})$$

- Structure Factor Equation:

$$\underline{F}(\underline{h}) = \Sigma_j f_j(\underline{s}) \exp(2\pi i \, \underline{h}^T \underline{u})$$

- Because:

$$\underline{s}^T \underline{x} = (\boldsymbol{F}^T \underline{h})^T (\boldsymbol{O} \underline{u}) = \underline{h}^T \boldsymbol{F} \boldsymbol{O} \underline{u} = \underline{h}^T \underline{u}$$

Coordinate Systems: Operators

Operators transform coordinates in such a way that a rigid body will moved to a new position within the coordinate system. We consider three types:

- Translation operators:
  - Move an object without rotating it.
- Rotation operators:
  - Rotate an object about the origin of the coordinate system.
- Rotation-translation (RT) operators:
  - Rotate and translate an object, or equivalently, rotate an object about a point other than the origin.

Coordinate Systems: Operators

Translation operators:

- Add the translation vector to the existing coordinate to get the new coordinate (in the same system).

$$\underline{x}_2 = \underline{x}_1 + \underline{T}_x$$

$$\underline{u}_2 = \underline{u}_1 + \underline{T}_u$$

- Translation vectors transform like coordinates, i.e. if $\underline{x}_1 = \boldsymbol{O} \underline{u}_1$ and $\underline{x}_2 = \boldsymbol{O} \underline{u}_2$ then:

$$\underline{T}_x = \boldsymbol{O} \underline{T}_u$$

$$\underline{T}_u = \boldsymbol{F} \underline{T}_x$$

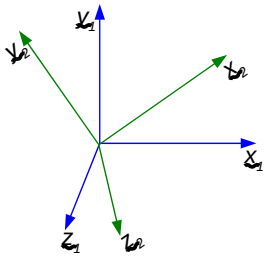
- The inverse of $\underline{T}$ is $-\underline{T}$.

Coordinate Systems: Operators

Rotation operators:

- A rotation is described by a matrix $\boldsymbol{R}$ which is orthonormal (i.e. $\boldsymbol{R}^{-1} = \boldsymbol{R}^{T}$)

  $\underline{x}_2 = \boldsymbol{R}_x \underline{x}_1$



$$\boldsymbol{R}_x = \begin{pmatrix} \cos(\sphericalangle x_1 x_2) & \cos(\sphericalangle x_1 y_2) & \cos(\sphericalangle x_1 z_2) \\ \cos(\sphericalangle y_1 x_2) & \cos(\sphericalangle y_1 y_2) & \cos(\sphericalangle y_1 z_2) \\ \cos(\sphericalangle z_1 x_2) & \cos(\sphericalangle z_1 y_2) & \cos(\sphericalangle z_1 z_2) \end{pmatrix}$$

Coordinate Systems: Operators

Rotation operators:

- We can represent the rotation in fractional coords:

  $\underline{x}_2 = \boldsymbol{R}_x \underline{x}_1$

  $\underline{u}_2 = \boldsymbol{R}_u \underline{u}_1$

  Since $\underline{x}_1 = \boldsymbol{O}\,\underline{u}_1$ and $\underline{x}_2 = \boldsymbol{O}\,\underline{u}_2$, $\boldsymbol{O}\,\underline{u}_2 = \boldsymbol{R}_x \boldsymbol{O}\,\underline{u}_1$

  therefore:

  $\boldsymbol{R}_u = \boldsymbol{O}^{-1} \boldsymbol{R}_x \boldsymbol{O}$

  $\boldsymbol{R}_x = \boldsymbol{O} \boldsymbol{R}_u \boldsymbol{O}^{-1}$

$$\boldsymbol{R}_u = \boldsymbol{O}^{-1} \boldsymbol{R}_x \boldsymbol{O}$$

$$\boldsymbol{R}_x = \boldsymbol{O} \boldsymbol{R}_u \boldsymbol{O}^{-1}$$

- Note: $\boldsymbol{R}_u$ is **not** a rotation matrix. ($\boldsymbol{R}_u^{-1} \neq \boldsymbol{R}_u^{T}$)

Coordinate Systems: Operators

Rotation-translation (RT) operators:

- A rotation-translation operator is described by a rotation matrix $\boldsymbol{R}$ followed by a translation $\underline{T}$.

  $\underline{x}_2 = \boldsymbol{R}_x \underline{x}_1 + \underline{T}_x$

- We can represent this as a single vector **operator**:

  $\underline{x}_2 = R_x(\underline{x}_1)$

- Its inverse is given by the rotation $\boldsymbol{R}_x^{-1}$ and the translation $-\boldsymbol{R}_x^{-1}\underline{T}_x$:

  $\underline{x}_1 = \boldsymbol{R}_x^{-1} (\underline{x}_2 - \underline{T}_x) = \boldsymbol{R}_x^{-1}\underline{x}_2 - \boldsymbol{R}_x^{-1}\underline{T}_x$

- To convert to fractional, convert $\boldsymbol{R}$ and $\underline{T}$ as before.

Coordinate Systems: Operators

Rotation-translation (RT) operators:

- Implementation:
  - Use a class to hold the rotation matrix and translation vector, and provide methods to transform a vector, invert, and convert the operator to other coordinate systems.
  - Historical: Fortran 77 does not support classes, so developers often grouped the rotation and translation in a 3x4 or 4x4 matrix. Mathematically, vectors must be augmented to length 4 by appending a '1'.

Coordinate Systems: Rotations

Rotations have many representations:

- Matrix:
  - use directly to manipulate vectors
  - uniquely defined, 9 numbers (m_{ij}).
- Quaternions:
  - uniquely defined, 4 numbers (x,y,z,w).
- Euler angles:
  - multiple conventions, 3 numbers (α,β,γ).
- Polar angles:
  - multiple conventions, 3 numbers (ϕ,ψ,κ).

Coordinate Systems: Rotations

Quaternions:

- 4 numbers: (x,y,z,w)
  - x,y,z are the direction cosines of the rotation axis, scaled by $sin(\kappa/2)$
  - w gives the angle of rotation, as $cos(\kappa/2)$.
- No ambiguity in definition.
- Easy to convert to Matrix, Euler, Polar
  - good as an interchange format. See **Clipper**, rotation.cpp

Coordinate Systems: Rotations

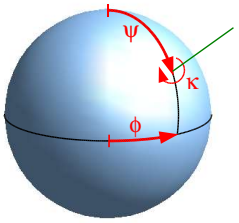
Euler angles:

- 3 numbers: (α,β,γ).
  - α is rotation about $\underline{z}$
  - β is rotation about new $\underline{y}$
  - γ is rotation about new $\underline{z}$
- 24 conventions (which axis to rotate about, stationary or moving axes), but crystallographers all uses ZYZr, so well standardised.
- Convenient for rotation function search limits.
- Convenient for program input.

Coordinate Systems: Rotations

Polar angles:

- 3 numbers: (ϕ,ψ,κ) or (ω,ϕ,κ)
- (ϕ,ψ) define the direction of the axis, κ is the angle of rotation about it.
- Easy to understand.
- Inconsistent conventions.
- Use for program output only.



Coordinate Systems: Derivatives

Many calculations require that we calculate derivatives of some function with respect to some coordinate. e.g.

- Refinement:
  - Refinement of individual atomic coordinates and B-factors: $(\underline{x}_i, B_i)$
- Molecular replacement:
  - Rigid body refinement of search model against density: $(\boldsymbol{R}_x, \underline{T}_x)$.

Coordinate Systems: Derivatives

- Refinement, MR both depend on calculation of density gradients and curvatures – these are calculated along grid/cell directions, i.e. fractional coordinates.
- Rigid body rotations, and refinement restraints (e.g. bond lengths/angles), are calculated using orthogonal coordinates.
- Need to convert derivatives between fractional and orthogonal.

Coordinate Systems: Derivatives

- e.g. for density gradients, $f = \rho$

$$\underline{g}_u = \begin{pmatrix} \frac{\partial f}{\partial u} \\ \frac{\partial f}{\partial v} \\ \frac{\partial f}{\partial w} \end{pmatrix} \quad \boldsymbol{C}_u = \begin{pmatrix} \frac{\partial^2 f}{\partial u^2} & \frac{\partial^2 f}{\partial u \partial v} & \frac{\partial^2 f}{\partial u \partial w} \\ \frac{\partial^2 f}{\partial v \partial u} & \frac{\partial^2 f}{\partial v^2} & \frac{\partial^2 f}{\partial v \partial w} \\ \frac{\partial^2 f}{\partial w \partial u} & \frac{\partial^2 f}{\partial w \partial v} & \frac{\partial^2 f}{\partial w^2} \end{pmatrix}$$

$$\underline{g}_x = \begin{pmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial z} \end{pmatrix} \quad \boldsymbol{C}_x = \begin{pmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial x \partial z} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} & \frac{\partial^2 f}{\partial y \partial z} \\ \frac{\partial^2 f}{\partial z \partial x} & \frac{\partial^2 f}{\partial z \partial y} & \frac{\partial^2 f}{\partial z^2} \end{pmatrix}$$

Coordinate Systems: Derivatives

- Gradients transform using the transpose of the inverse matrix (because the coordinate is in the denominator):

$$\underline{g}_u = \boldsymbol{O}^T \underline{g}_x$$
$$\underline{g}_x = \boldsymbol{F}^T \underline{g}_u$$
$$g_{u,j} = \Sigma_i O_{ij} g_{x,i} \qquad g_{x,j} = \Sigma_i F_{ij} g_{u,i}$$

- Curvatures:

$$\boldsymbol{C}_u = \boldsymbol{O}^T \boldsymbol{C}_x \boldsymbol{O}$$
$$\boldsymbol{C}_x = \boldsymbol{F}^T \boldsymbol{C}_u \boldsymbol{F}$$
$$C_{u,kl} = \Sigma_i \Sigma_j O_{ik} O_{jl} C_{x,ij} \qquad C_{x,kl} = \Sigma_i \Sigma_j F_{ik} F_{jl} C_{u,ij}$$

Coordinate Systems

Summary:

- In crystallographic calculations, we need to use a range of coordinate systems:
  - real and reciprocal space
  - orthogonal, fractional, and grid.
- We also use operators in each space.
  - rotations, translations, and RT.
- Coordinates and operators are related by orthogonalising and fractionalising matrices and their transposes in various combinations.