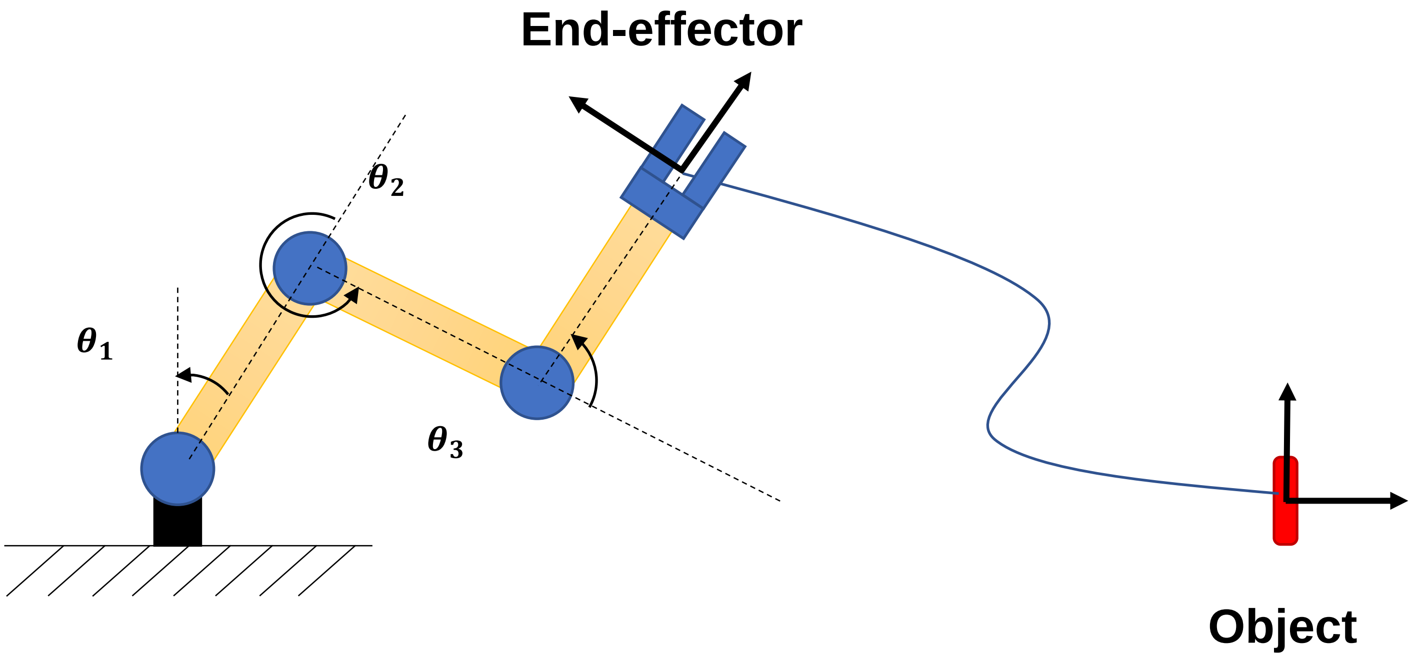
Velocities and the Robot Jacobians

Topics

- Velocities in Cartesian space
- Jacobian matrices

Complementary Reading: J.J. Craig, Introduction to Robotics, Chapter 5,9.

Motivating Example



High-level goal: move robot to grasp the object

- Trajectory could be straight line
- Trajectory could be curved line (avoid obstacles, satisfy constraints, etc.)
- In both situations, often better to define the trajectory in Cartesian Space
  - How then do we determine $\{\dot{\mathrm{p}}(t), \dot{R}(t)\} \rightarrow \dot{\theta}(t)$?

First, need to better define position and angular velocities.

Translational and Rotational Velocities

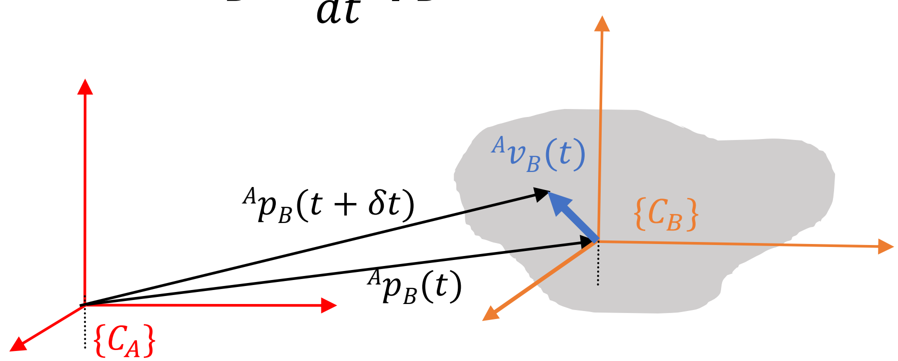
Velocities are vectors, which means they are expressed w.r.t. coordinate frames.

- Velocity vectors are *free vectors,* i.e., they do not have an "origin"
- Are described simply by the basis vectors of the coordinate frame
- Two types of velocities, **rotational** *and* **translational**.

Translation Velocities

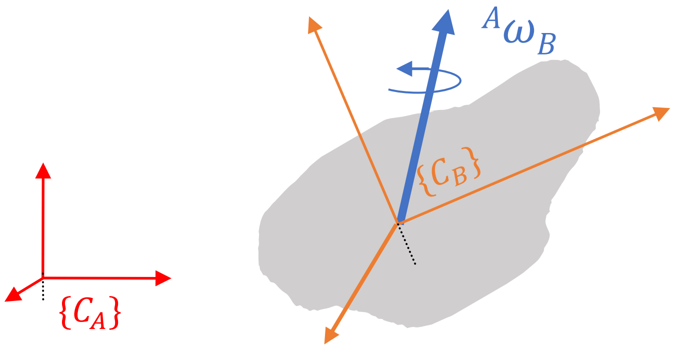
- Describe how objects/frames move *without rotation* in space.

$$Av_B = \frac{d}{dt}{}^A p_B$$



Rotational Velocities

- The angular velocity, or change in the orientation, $\dot{R}$, has a special form.
  - Cannot simply take $\frac{d}{dt}$ of the elements of R



- Frame B has an orientation ${}^{A}_{B}R$ and its rotational motion may be represented by the rotational (angular) velocity vector ${}^{A}\omega_{B} \in \mathbb{R}^3$
  - $||{}^{A}\omega_{B}||$ defines speed of rotation
  - ${}^{A}\omega_{B}/||{}^{A}\omega_{B}||$: defines axis of rotation (defined in $\{A\}$)

Definitions of angular velocity with $\dot{R}$

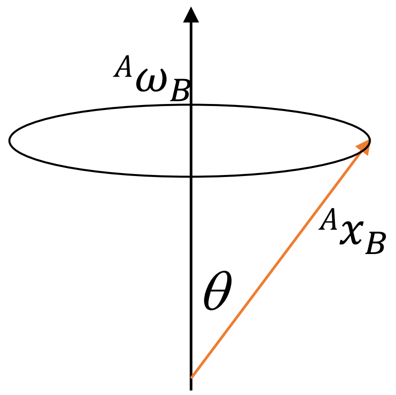
- The angular velocity described in orientation is defined by the **change in basis vectors**

$$\dot{R} = \left[{}^{A}\dot{u}_x, {}^{A}\dot{u}_y, {}^{A}\dot{u}_z\right]$$

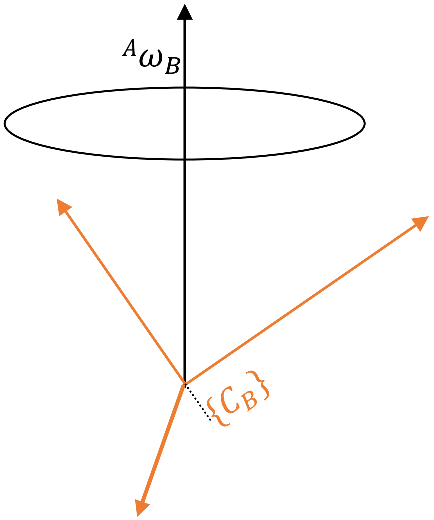
- Consider coordinate $\{C_B\}$ and its basis vector about x. If it processes about ${}^{A}\omega_B$, then the change in ${}^{A}u_{x,b}$ has…
  - Magnitude: $\|{}^{A}\omega_B\|\left\|{}^{A}u_{x,B}\right\| \sin\theta = \|{}^{A}\omega_B\| \sin\theta$
  - Direction: perpendicular to plane spanned by ${}^{A}\omega_B, {}^{A}u_{x,B}$

- Thus, it is exactly the cross product

$${}^{A}\dot{x}_B = {}^{A}\omega_B \times {}^{A}u_{x,B}$$



- Therefore, for each basis vector,



$$ {}^{A}\dot{u}_{x,B} = {}^{A}\omega_B \times {}^{A}u_{x,B} = {}^{A}\widehat{\omega}_B {}^{A}u_{x,B} $$
$$ {}^{A}\dot{u}_{y,B} = {}^{A}\omega_B \times {}^{A}u_{y,B} = {}^{A}\widehat{\omega}_B {}^{A}u_{y,B} $$
$$ {}^{A}\dot{u}_{z,B} = {}^{A}\omega_B \times {}^{A}u_{z,B} = {}^{A}\widehat{\omega}_B {}^{A}u_{z,B} $$

Can be written in matrix form, i.e. $\dot{R}$

Let $\widehat{\omega}$ be

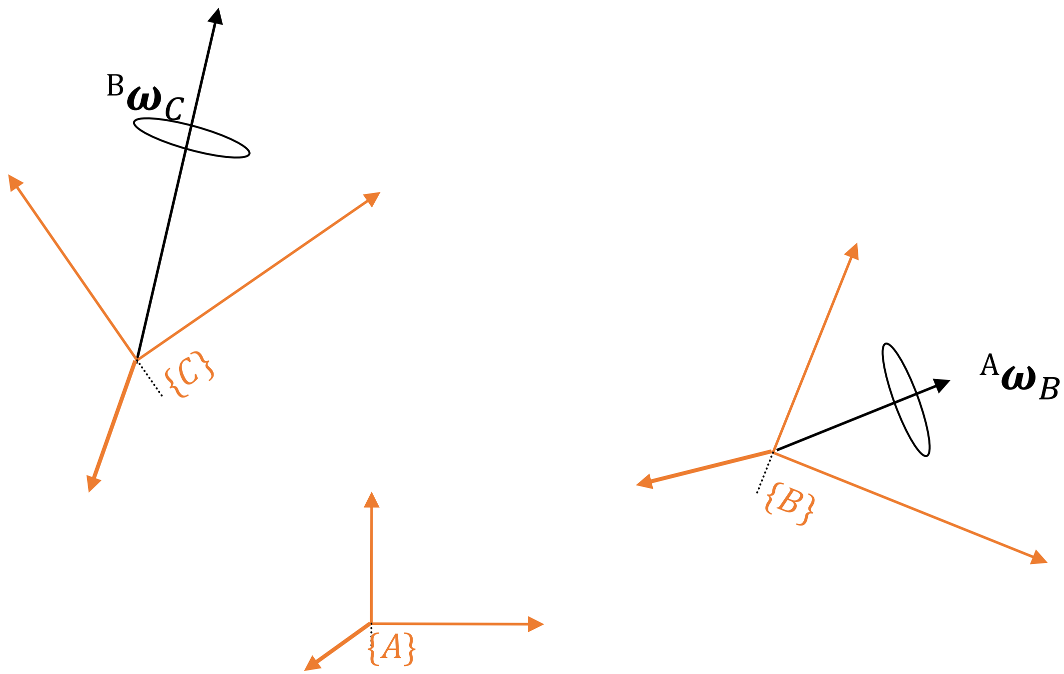
$$ \widehat{\omega} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} $$

Given $\dot{R} = \left[{}^{A}\dot{u}_x, {}^{A}\dot{u}_y, {}^{A}\dot{u}_z\right]$ and ${}^{A}_{B}R = \left[{}^{A}u_{x,B}, {}^{A}u_{y,B}, {}^{A}u_{z,B}\right]$, Then

$$ {}^{A}\dot{R}_B = {}^{A}\widehat{\omega}_B \, {}^{A}_{B}R $$

- Describing angular velocities in different coordinate frames is quite straightforward:

$$^{A}\omega_C = {}^{A}\omega_B + {}^{A}_{B}R\,{}^{B}\omega_C$$



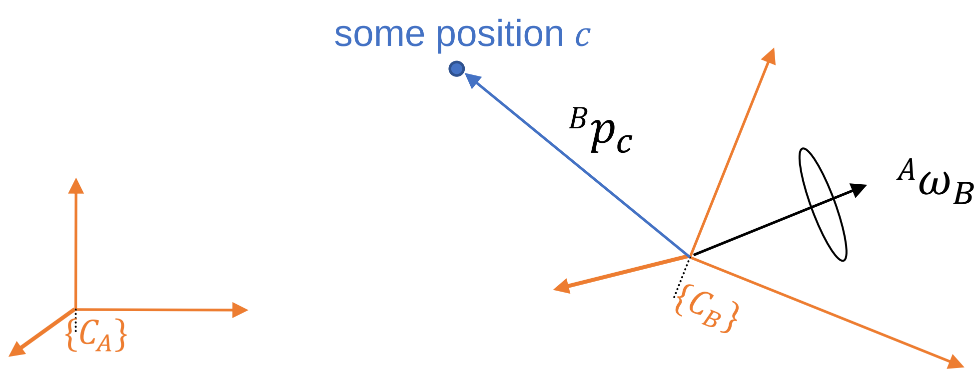
- Describing angular velocities using $\omega \in \mathbb{R}^3$ is useful for control.

Describing Translational Velocity in Different Frames

- More complicated since rotating coordinate frames play a role.
- Find by differentiating position:

$$^{A}p_C = {}^{A}p_B + {}^{A}_{B}R \, {}^{B}p_C$$

$$\begin{aligned} {}^{A}v_C = {}^{A}\dot{p}_C &= {}^{A}\dot{p}_B + \frac{d}{dt}\left({}^{A}_{B}R \, {}^{B}p_C\right) \\ &= {}^{A}v_B + {}^{A}_{B}R \, {}^{B}\dot{p}_C + {}^{A}_{B}\dot{R} \, {}^{B}p_C \\ &= {}^{A}v_B + {}^{A}_{B}R \, {}^{B}v_C + {}^{A}\widehat{\omega}_B \, {}^{A}_{B}R \, {}^{B}p_C \end{aligned}$$



Translational and Angular Accelerations

For equations of acceleration, simply differentiate the former velocity equations.

Rotational Acceleration:

$$^{A}\omega_{C} = {}^{A}\omega_{B} + {}^{A}_{B}R\,{}^{B}\omega_{C}$$

$$^{A}\alpha_{C} = {}^{A}\dot{\omega}_{C} = {}^{A}\dot{\omega}_{B} + \frac{d}{dt}\left({}^{A}_{B}R\,{}^{B}\omega_{C}\right)$$

$$= {}^{A}\alpha_{B} + {}^{A}_{B}\dot{R}\,{}^{B}\omega_{C} + {}^{A}_{B}R\,{}^{B}\dot{\omega}_{C}$$

$$= {}^{A}\alpha_{B} + {}^{A}\widehat{\omega}_{B}\,{}^{A}_{B}R\,{}^{B}\omega_{C} + {}^{A}_{B}R\,{}^{B}\alpha_{C}$$

Translational Acceleration:

$$^{A}v_{C} = {}^{A}v_{B} + {}^{A}_{B}R\,{}^{B}v_{C} + {}^{A}\widehat{\omega}_{B}\,{}^{A}_{B}R\,{}^{B}p_{C}$$

$$^{A}a_{C} = {}^{A}\dot{v}_{C} = {}^{A}\dot{v}_{B} + {}^{A}_{B}\dot{R}\,{}^{B}v_{C} + {}^{A}_{B}R\,{}^{B}\dot{v}_{C} + {}^{A}\dot{\widehat{\omega}}_{B}\,{}^{A}_{B}R\,{}^{B}p_{C} + {}^{A}\widehat{\omega}_{B}\,{}^{A}_{B}\dot{R}\,{}^{B}p_{C} + {}^{A}\widehat{\omega}_{B}\,{}^{A}_{B}R\,{}^{B}\dot{p}_{C}$$

$$= {}^{A}a_{B} + {}^{A}\widehat{\omega}_{B}\,{}^{A}_{B}R\,{}^{B}v_{C} + {}^{A}_{B}R\,{}^{B}a_{C} + {}^{A}\dot{\widehat{\omega}}_{B}\,{}^{A}_{B}R\,{}^{B}p_{C} + {}^{A}\widehat{\omega}_{B}\,{}^{A}_{B}\dot{R}\,{}^{B}p_{C} + {}^{A}\widehat{\omega}_{B}\,{}^{A}_{B}R\,{}^{B}v_{C}$$

$$= {}^{A}a_{B} + 2\,{}^{A}\widehat{\omega}_{B}\,{}^{A}_{B}R\,{}^{B}v_{C} + {}^{A}_{B}R\,{}^{B}a_{C} + {}^{A}\widehat{\alpha}_{B}\,{}^{A}_{B}R\,{}^{B}p_{C} + {}^{A}\widehat{\omega}_{B}\,{}^{A}\widehat{\omega}_{B}\,{}^{A}_{B}R\,{}^{B}p_{C}$$

Notes:

- Consider always apply the general form
- *Very easy* to make a mistake if you forgot some terms if doing differentiation on a robot kinematics
  - e.g. coordinate frame orientations line up and ${}^{A}_{B}R = I$ and you omit R during differentiation
  - e.g. coordinate frame origins line up and ${}^{\mathrm{A}}p_B = 0$ and you omit ${}^{\mathrm{A}}p_B$ during differentiation

The Jacobian Matrix

- Now we are ready to describe the relationship between the joint velocities and the end effector velocities (both translational and angular), using the *Jacobian matrix.*

Deriving the Jacobian Matrix from Forward Kinematics:

$$[x, y, z, r, p, \gamma] \stackrel{\text{def}}{=} \chi = f(\theta) \stackrel{\text{def}}{=} \begin{bmatrix} f_1(\theta_1 \dots \theta_n) \\ \vdots \\ f_6(\theta_1 \dots \theta_n) \end{bmatrix}$$

Expand using Taylor series:

$$x = f_1(\theta + \delta\theta) \approx f_1(\theta) + \left[\frac{\partial f_1(\theta)}{\partial \theta_1} \dots \frac{\partial f_1(\theta)}{\partial \theta_n}\right] [\delta\theta_1 \dots \delta\theta_n]^\top + O(\theta^2)$$

$$f_1(\theta + \delta\theta) - f_1(\theta) = \left[\frac{\partial f_1(\theta)}{\partial \theta_1} \dots \frac{\partial f_1(\theta)}{\partial \theta_n}\right] [\delta\theta_1 \dots \delta\theta_n]^\top + O(\theta^2)$$

$$\dot{x} \approx \left[\frac{\partial f_1(\theta)}{\partial \theta_1} \dots \frac{\partial f_1(\theta)}{\partial \theta_n}\right] \dot{\theta} + O(\theta^2)$$

$$\approx J_1(\theta)\dot{\theta}$$

- Assume that we have an n-link robot with joint variables $\theta = \theta_1, \theta_2, \dots, \theta_N$

$$^0\chi_N = {}^0[x, y, z, r, p, \gamma]_N = f(\theta)$$

- If we differentiate $^0\chi_N = f(\theta)$ over time,

$$^0\dot{\chi}_N = [{}^0\dot{p}_N, {}^0\omega_N]^\top = \dot{f}(\theta)$$

and

$$^0\dot{\chi}_N = [{}^0\dot{p}_N, {}^0\omega_N]^\top \approx J(\theta)\dot{\theta}$$

where

$$J(q) = \begin{bmatrix} \frac{\partial f_1(\theta)}{\partial \theta_1} & \cdots & \frac{\partial f_1(\theta)}{\partial q_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_6(\theta)}{\partial \theta_1} & \cdots & \frac{\partial f_6(\theta)}{\partial \theta_N} \end{bmatrix}$$

is the Jacobian matrix

What does this Jacobian mapping mean?

$$J(q) = \begin{bmatrix} \frac{\partial f_1(\theta)}{\partial \theta_1} & \cdots & \frac{\partial f_1(\theta)}{\partial q_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_6(\theta)}{\partial \theta_1} & \cdots & \frac{\partial f_6(\theta)}{\partial \theta_N} \end{bmatrix}$$

- **Each row describes** the *weighted effect of every joint's velocities* on one axis of velocity ($x, y, z, r, p,$ or γ) of the end effector
- **Each column describes** the *effect* of *one joint's velocities* on every axis of velocity ($x, y, z, r, p,$ and γ) of the end effector

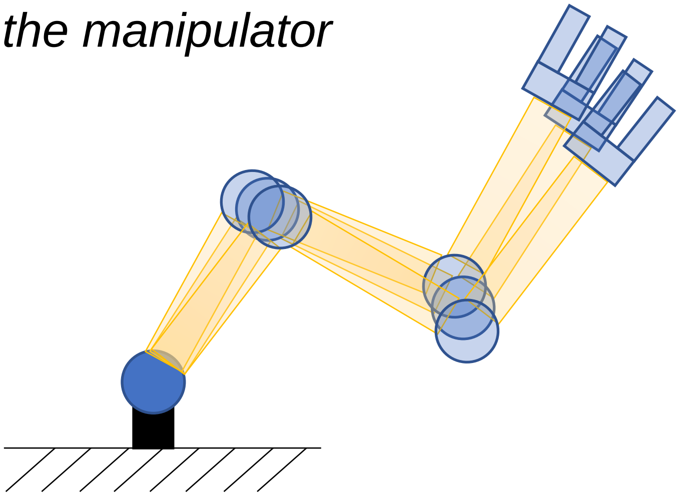
Most importantly, through the *Jacobian matrix inverse* J^{-1}, one can determine how to the joint angles should change to produce a change in output pose:

$$\begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_N \end{bmatrix} = J^{-1}(\theta) \begin{bmatrix} {}^0\dot{p}_N \\ {}^0\omega_N \end{bmatrix}$$

Using the Jacobian Matrix for Control

The Jacobian matrix is a *linear approximation of the manipulator*

- i.e. valid in a close proximity $\theta \pm \delta$.



- Linear control can be used:

  Forward Kinematics:

  $$x = f(\theta)$$

  $$x + \pm\delta x = f(\theta + \delta\theta)$$

  Approximation used for linear control:

  $$x \pm \delta x \approx f(\theta) + J(\theta)\delta\theta$$

  $$\delta x \approx J(\theta)\delta\theta$$

  Given a desired incremental desired δx, apply

  $$\delta\theta = J(\theta)^{-1}\delta x$$

Key insights

$$\begin{bmatrix} {}^0\dot{p}_N \\ {}^0\omega_N \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1(\theta)}{\partial \theta_1} & \cdots & \frac{\partial f_1(\theta)}{\partial \theta_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_6(\theta)}{\partial \theta_1} & \cdots & \frac{\partial f_6(\theta)}{\partial \theta_N} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \vdots \\ \dot{\theta}_N \end{bmatrix}$$

- This linear controller $\delta\theta = J(\theta)^{-1}\delta x$ involves solving the matrix inverse problem
  - If $n > 6$, fat matrix J, typically an underconstrained problem.
  - If $n < 6$, skinny matrix J, typically an overconstrained problem.
  - If $n = 6$, exact solution.
  - (the above assumes full rank J)

Pseudoinverse of Matrices

Pseudoinverse is used in practice for matrix inverses (not only for Jacobians), to avoid rank-deficient matrix inversion.

- **Standard pseudo-inverse** (not advised)

$$J^{\dagger} = J^{\top}(J^{\top}J)^{-1} \qquad \text{or} \qquad J^{\dagger} = J^{\top}(JJ^{\top})^{-1}$$

  - Depends on fat or thin J

- **Robust pseudo-inverse method 1: Damped Least Squares**

$$J^{\dagger} = J^{\top}(J^{\top}J + \lambda^2 I)^{-1} \qquad \text{or} \qquad J^{\dagger} = J^{\top}(JJ^{\top} + \lambda^2 I)^{-1}$$

- **Robust pseudo inverse method 2: Singular Value Decomposition**

$$J = U\Sigma V^{\top} \;\rightarrow\; J^{\dagger} = V\Sigma^{-1}U^{\top}$$

  - where $\Sigma^{-1} = diag(\sigma_{max}, \ldots, \sigma_i, \ldots, \sigma_{min})$
  - where if $\frac{\sigma_{max}}{\sigma_i} > \mathrm{max\_ratio}$ then $\sigma_{i\ldots min} = 0$.