Robot Kinematics

Jiayuan Gu

Slides prepared by Prof. Hao Su with the help of
Yuzhe Qin, Minghua Liu, Fanbo Xiang, Jiayuan Gu

Agenda

- Kinematics Equations
- Forward Kinematics
    - Jacobian of Kinematic Chain
- Inverse Kinematics
- Screw and Twist

Kinematics Equations

Kinematics Equations

- “Define how **input movement** at one or more joints specifies the configuration of the device, in order to **achieve a task position** or end-effector location.”

- Map the joint space coordinate $\theta \in \mathbb{R}^n$ to a transformation matrix T:

$$T_{s \to e} = f(\theta)$$

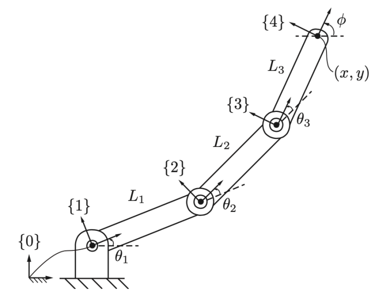
- Calculated by composing transformations along the kinematic chain

https://en.wikipedia.org/wiki/Kinematics_equations

Kinematics Equations

- The kinematics equations of a serial chain of n links, with joint parameters θ_i are given by

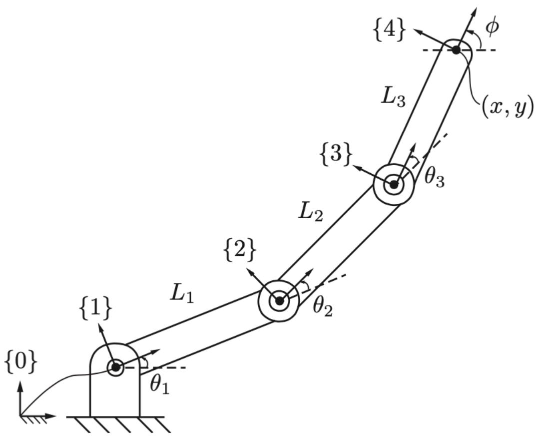
$$T = \prod_{i=1}^{n} Z_i X_i$$



- Joint matrices $Z_i(\theta_i)$ characterize the relative movement at each joint
- Link matrices $X_i(\theta_i)$ define the geometry of each link

https://en.wikipedia.org/wiki/Kinematics_equations

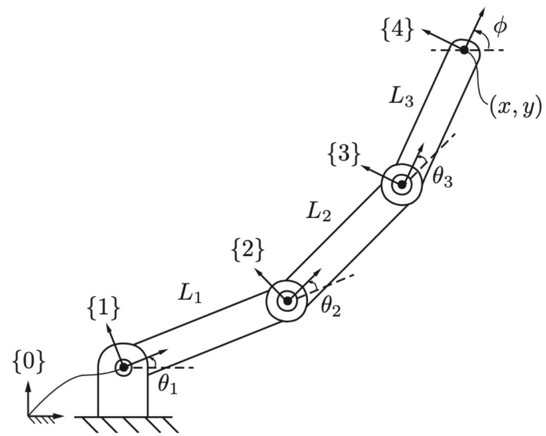
Forward Kinematic Problem



- “Forward kinematics refers to the use of the kinematic equations of a robot to compute the position of the end-effector from specified values for the joint parameters.”
- Given θ, what is $T_{s \rightarrow e} = f(\theta)$?

https://en.wikipedia.org/wiki/Forward_kinematics

Forward Kinematic Problem



- Given θ, what is $T_{s \to e} = f(\theta)$?
- Given θ and $\Delta\theta$, what is $T_{s \to e(\theta + \Delta\theta)} = f(\theta + \Delta\theta)$?
- Given $\theta(t)$, what is $\dot{T}_{s \to e(\theta)} = \dot{f}(\theta)\dot{\theta}$?

What is $\dot{T}_{s\to e}$?

- Derivative of $T_{s\to e} \in \mathbb{SE}(3)$

- Checking the differential:

$$T^o_{s\to e(t+\Delta t)} - T^o_{s\to e(t)} = T^o_{e(t)\to e(t+\Delta t)} T^o_{s\to e(t)} - T^o_{s\to e(t)}$$

(using composition rule as linear transformation)

- $\dot{T}^o_{s\to e} := \lim_{\Delta t \to 0} \dfrac{T^o_{s\to e(t+\Delta t)} - T^o_{s\to e(t)}}{\Delta t}$

Representation of $\dot{T}_{s\to e}$

- Since $T_{s\to e} \in \mathbb{SE}(3)$ can be represented by a 4x4 matrix, $\dot{T}_{s\to e}$ can also be represented by a 4x4 matrix

- Are there any structures of $T_{s\to e}$ and $\dot{T}_{s\to e}$?

$\dot{T}_{s\rightarrow e}$, Screw, Twist

- We will introduce later
  - a 6D vector "**screw**" χ to describe the rigid transformation, so that $T = e^{\chi}$
  - a 6D vector "**twist**" ξ to describe the instant velocity

$\dot{T}_{s \to e}$ and Jacobian

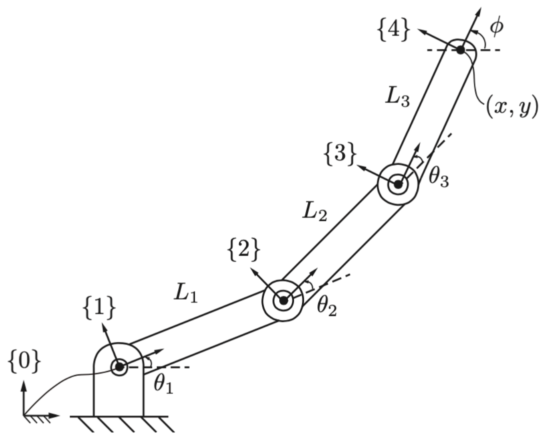
- In vector calculus, the **Jacobian** matrix of a vector-valued function of several variables is the matrix of all its first-order partial derivatives.

$$\mathbf{J} = \begin{bmatrix} \dfrac{\partial \mathbf{f}}{\partial x_1} & \cdots & \dfrac{\partial \mathbf{f}}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \nabla^{\mathrm{T}} f_1 \\ \vdots \\ \nabla^{\mathrm{T}} f_m \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f_1}{\partial x_1} & \cdots & \dfrac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial f_m}{\partial x_1} & \cdots & \dfrac{\partial f_m}{\partial x_n} \end{bmatrix}$$

- Given $\dot{T}_{s \to e(\theta)} = \dot{f}(\theta)\dot{\theta}, \dot{f}(\theta)$ is kind of a Jacobian matrix

https://en.wikipedia.org/wiki/Jacobian_matrix_and_determinant

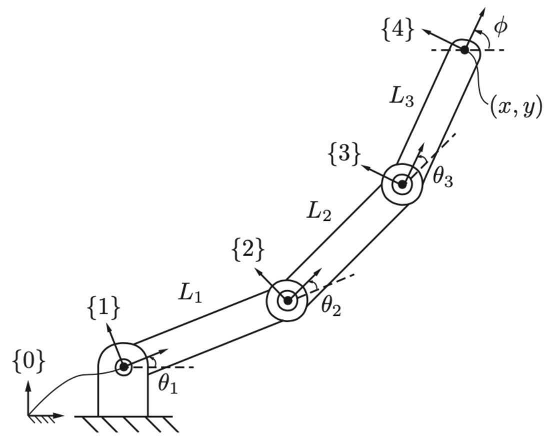
Inverse Kinematic Problem



- "Inverse kinematics makes use of the kinematics equations to determine the joint parameters that provide a desired configuration (position and rotation) for the end-effector."
- Given $T_{s \to e}$, what is θ by solving $T_{s \to e} = f(\theta)$?

https://en.wikipedia.org/wiki/Inverse_kinematics

Inverse Kinematic Problem



- Given $T_{s \to e}$, what is θ by solving $T_{s \to e} = f(\theta)$?
- Given θ and $T_{s \to e(\theta + \Delta\theta)}$, what is $\Delta\theta$ by solving $T_{s \to e(\theta + \Delta\theta)} = f(\theta + \Delta\theta)$?
- Given $\dot{T}_{s \to e(\theta)}$, what is $\dot{\theta}(t)$ by solving $\dot{T}_{s \to e(\theta)} = \dot{f}(\theta)\dot{\theta}$?

https://en.wikipedia.org/wiki/Inverse_kinematics

Two Types of Approaches

- Analytical Solution
  - Compute the inverse mapping of $T_{s \to e} = f(\theta)$
- Numerical Solution
  - Solve $T_{s \to e} = f(\theta)$ by numerical methods using gradients (Jacobian) $\dot{f}(\theta)$

Jacobian of Kinematic Chain

Geometric Jacobian

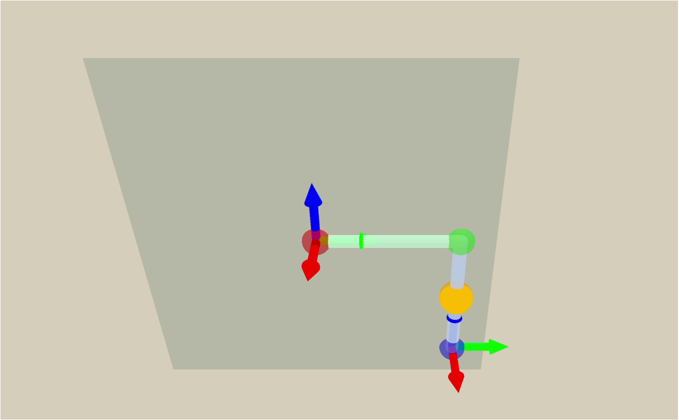
- Kinematics Equation: $\dot{T}_{s\to e(t)} = \dot{f}(\theta)\dot{\theta}$

- There is a "minimal" representation of velocity, **twist** $\xi_{e(t)} \in \mathbb{R}^6$, such that $\dot{T}_{s\to e(t)} = g(\xi_{e(t)})T_{s\to e(t)}$, where $g : \mathbb{R}^6 \mapsto \mathbb{R}^{4\times 4}$ is a differentiable mapping

- In this section, we will discuss $\xi_{e(t)} = J(\theta)\dot{\theta}$

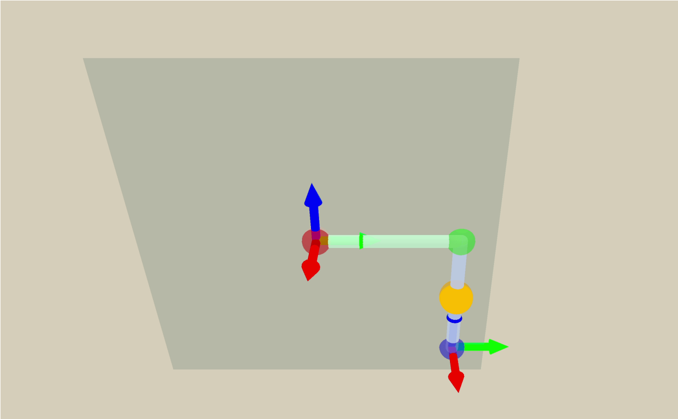
Example

$$T^{s}_{s\to b(t)} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha t) & -\sin(\alpha t) & 1+\sin(\alpha t) \\ 0 & \sin(\alpha t) & \cos(\alpha t) & -\cos(\alpha t) \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Example

$$\dot{T}^{s}_{s \to b(t)} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & -\sin(\alpha t) & -\cos(\alpha t) & \cos(\alpha t) \\ 0 & \cos(\alpha t) & -\sin(\alpha t) & \sin(\alpha t) \\ 0 & 0 & 0 & 0 \end{bmatrix} \alpha$$



Geometric Jacobian

- Recall
  - $\dot{T}^o_{s\to e} := \lim_{\Delta t \to 0} \frac{T^o_{s\to e(t+\Delta t)} - T^o_{s\to e(t)}}{\Delta t}$
- Two commonly used observer frames:
  - Spatial twist $\xi^s_{e(t)}$
  - Body twist $\xi^b_{e(t)}$ when $b = e(t)$

Spatial Geometric Jacobian

- Spatial Geometric Jacobian $J^s(\theta)$:

$$\xi^s_{e(t)} = J^s(\theta)\dot{\theta}$$

where $\theta \in \mathbb{R}^n$ (n joints), $J^s(\theta) \in \mathbb{R}^{6\times n}$

- The i-th column of $J(\theta)$ is ${}^i\hat{\xi}^s_{e(t)}$, the twist when the movement is caused only by the i-th joint **while all other joints stay static**

Spatial Geometric Jacobian

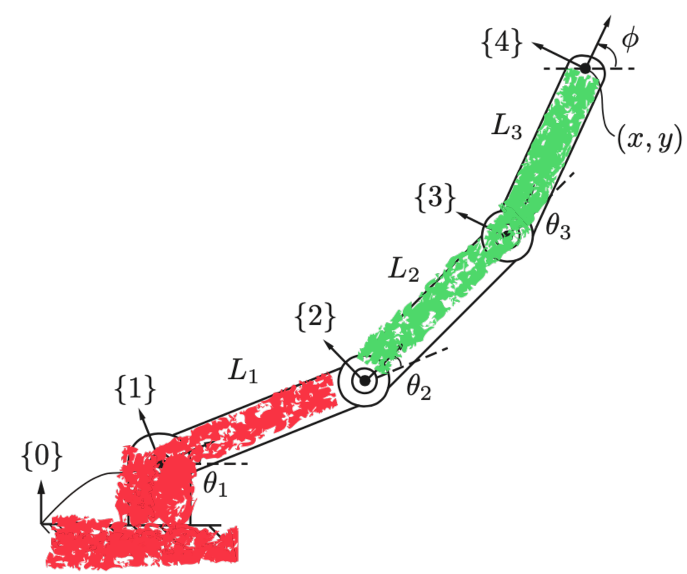
- Spatial Geometric Jacobian $J^s(\theta)$:

$$\xi^s_{e(t)} = J^s(\theta)\dot{\theta}$$

where $\theta \in \mathbb{R}^n$ (n joints), $J^s(\theta) \in \mathbb{R}^{6\times n}$

- The i-th column of $J(\theta)$ is ${}^{i}\hat{\xi}^s_{e(t)}$, the twist when the movement is caused only by the i-th joint **while all other joints stay static**

- For example, ${}^{2}\hat{\xi}^s_{e(t)}$ describes the motion of the green part, which is to revolute about Joint {2}



Body Geometric Jacobian

- Body Geometric Jacobian $J^b(\theta)$:

$$\xi^b_{e(t)} = J^b(\theta)\dot{\theta}$$

where $J^b(\theta) \in \mathbb{R}^{6\times n}$

- The i-th column of $J(\theta)$ is ${}^{i}\hat{\xi}^b_{e(t)}$, the twist when the movement is caused only by the i-th joint **while all other joints stay static**

Body Geometric Jacobian

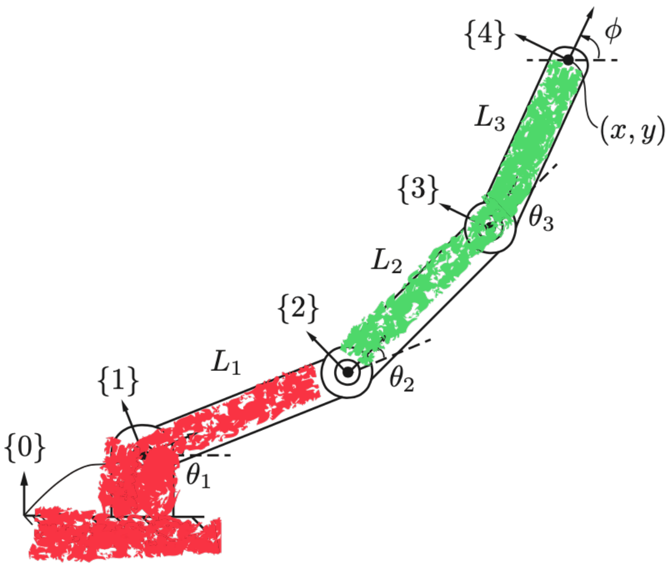
- Body Geometric Jacobian $J^b(\theta)$:

$$\xi^b_{e(t)} = J^b(\theta)\dot{\theta}$$

  where $J^b(\theta) \in \mathbb{R}^{6\times n}$

- The i-th column of $J(\theta)$ is ${}^i\hat{\xi}^b_{e(t)}$, the twist when the movement is caused only by the i-th joint **while all other joints stay static**

- For example, ${}^2\hat{\xi}^b_{e(t)}$ describes the motion of the green part observed by $\mathscr{F}_s = \mathscr{F}_{\{0\}}$, which is to revolute about Joint {2}



More about Jacobian

- Several libraries provide the computation of geometric Jacobian (e.g., pinocchio, pytorch_kinematics, polymetis)

- Geometric Jacobian $J(\theta) \in \mathbb{R}^{6\times n}$ usually refers to the mapping from **joint velocities** to **twist**

Inverse Kinematics

Inverse Kinematics

- Position query
  - Given the forward kinematics $T_{s \to e}(\theta)$ and the target pose $T_{target} = \mathbb{SE}(3)$, find θ that satisfies $T_{s \to e}(\theta) = T_{target}$
- Velocity query
  - Given the end-effector velocity (twist), find the joint velocity that satisfies $\xi_{target} = J(\theta)\dot{\theta}$
- May have multiple solutions, a unique solution or no solution

Null Space of Jacobian

- Consider the velocity query IK task
- Recall that $\xi = J(\theta)\dot{\theta}$ for an n-joint kinematic chain, where J is a $6 \times n$ matrix
- When $n > 6$, the joint space is projected to a lower-dimensional space and J must exist a null space
- As a result, IK may have infinite solutions (a special solution + any vector in the null space of J)
- The null space adds flexibility to make motion plans

Analytical Solution

- Try to solve the equation $T_{target} = T(\theta)$ and get an analytical solution for θ
  - e.g., solve θ_1 and θ_2 for $\begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & -\sin\theta_1(l_2 + l_3) \\ \sin\theta_1 & \cos\theta_1 & 0 & \cos\theta_1(l_2 + l_3) \\ 0 & 0 & 1 & l_1 - l_4 + \theta_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} = T_{target}$
- For robots with more than 3-DoF, analytical solution can be very complex
  - e.g., for a 6-DoF robot, you will need several pages to write down the formula
- Some useful libraries: IKFast, IKBT

Numerical Solution

- Solving a nonlinear optimization problem
- Standard numerical optimization algorithms can be utilized, e.g. Newton-Raphson and Levenberg-Marquardt
- Numerical IK leverages the geometric Jacobian
  $\xi = J(\theta)\dot{\theta}$

Levenberg–Marquardt Algorithm

- Error between the desired pose and the current one:

$$T_{err}(\theta) = T(\theta)T_{target}^{-1} \in \mathbb{SE}(3)$$

- Differentiate: $\dot{T}_{err}(\theta) = J_{err}(\theta)\dot{\theta}$

- There is a "minimal" representation, **screw** $\chi \in \mathbb{R}^6$, such that $\chi = G(T(\theta))$, where $G : \mathbb{R}^{4\times4} \mapsto \mathbb{R}^6$ is a differentiable mapping

Levenberg–Marquardt Algorithm

- In LM algorithm, we iteratively update θ
- In each iteration, we try to find a $\Delta\theta$ that minimizes:

$$S(\theta, \Delta\theta) = ||\chi_{err} + J_{err}(\theta)\Delta\theta||^2 + \lambda||\Delta\theta||^2$$

- λ term stabilizes the optimization
- Closed-form solution ($J = -J_{err}$):

$$(J^{\mathrm{T}}J + \lambda I)\Delta\theta = J^{\mathrm{T}}\chi_{err}$$

- Solve $\Delta\theta$ and then update θ by: $\theta \leftarrow \theta + \Delta\theta$

Levenberg–Marquardt Algorithm

$$(J^{\mathrm{T}}J + \lambda I)\Delta\theta = J^{\mathrm{T}}\chi_{err}$$

- Damping factor $\lambda \geq 0$ is adjusted at each iteration:
- If $S(\theta, \Delta\theta)$ is decreasing, a smaller λ (e.g., $\lambda \leftarrow 0.1\lambda$) can be used.
  - closer to the Gauss–Newton algorithm
- Otherwise, a larger λ (e.g., $\lambda \leftarrow 10\lambda$) can be used.
  - closer to the gradient-descent algorithm

Read by Yourself

Levenberg–Marquardt Algorithm

- LM algorithm may converge to a local minima, initial θ_0 is very important:
  - Sampling multiple θ_0 may boost the performance
- In most cases, θ comes with limit constraints:
  - $l[i] \leq \theta[i] \leq r[i]$
  - A joint can only translate (or rotate) within the limit
  - Invalid state rejection
  - Clipping during the optimization iterations

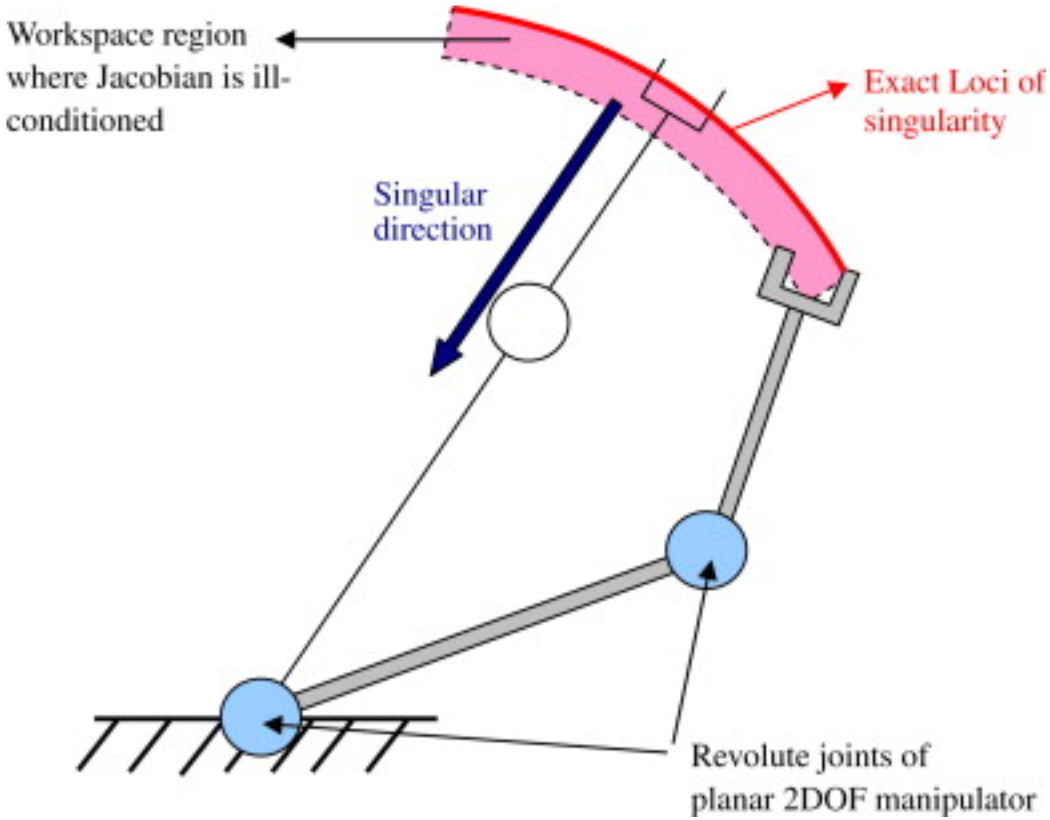
Read by Yourself

Kinematic Singularity

Question: Is it always possible to move the end-effector to any direction $\hat{\xi}$ for a robot with $\mathrm{DoF} \geq 6$?

- **Kinematic singularity:**
  - A **robot configuration** where the robot's end-effector loses the ability to move in one direction instantaneously
- If $\mathrm{rank}(J(\theta)) < 6$ at some θ, by $\Delta\xi = J(\theta)\Delta\theta$, $\Delta\xi$ can only be in a linear space with dimension $\mathrm{rank}(J(\theta)) < 6$, losing its ability to move in some directions
- Note: Kinematic singularity does not mean that there exists a configuration that is not accessible (may get to the pose by some other motion trajectory)

Kinematic Singularity



https://www.sciencedirect.com/science/article/pii/S0736584507001068