Agent-based Modeling for Hunter-Gatherer Communities - Day2: Modeling and Implementation of Interactions

Johannes Zonker

Hackathon on Small Data Analysis

January 2024

Outline

Interaction potential definition

- We introduce an interaction force based on the distance d between two agents and the function f_U on $\mathbb{R}$ defined by

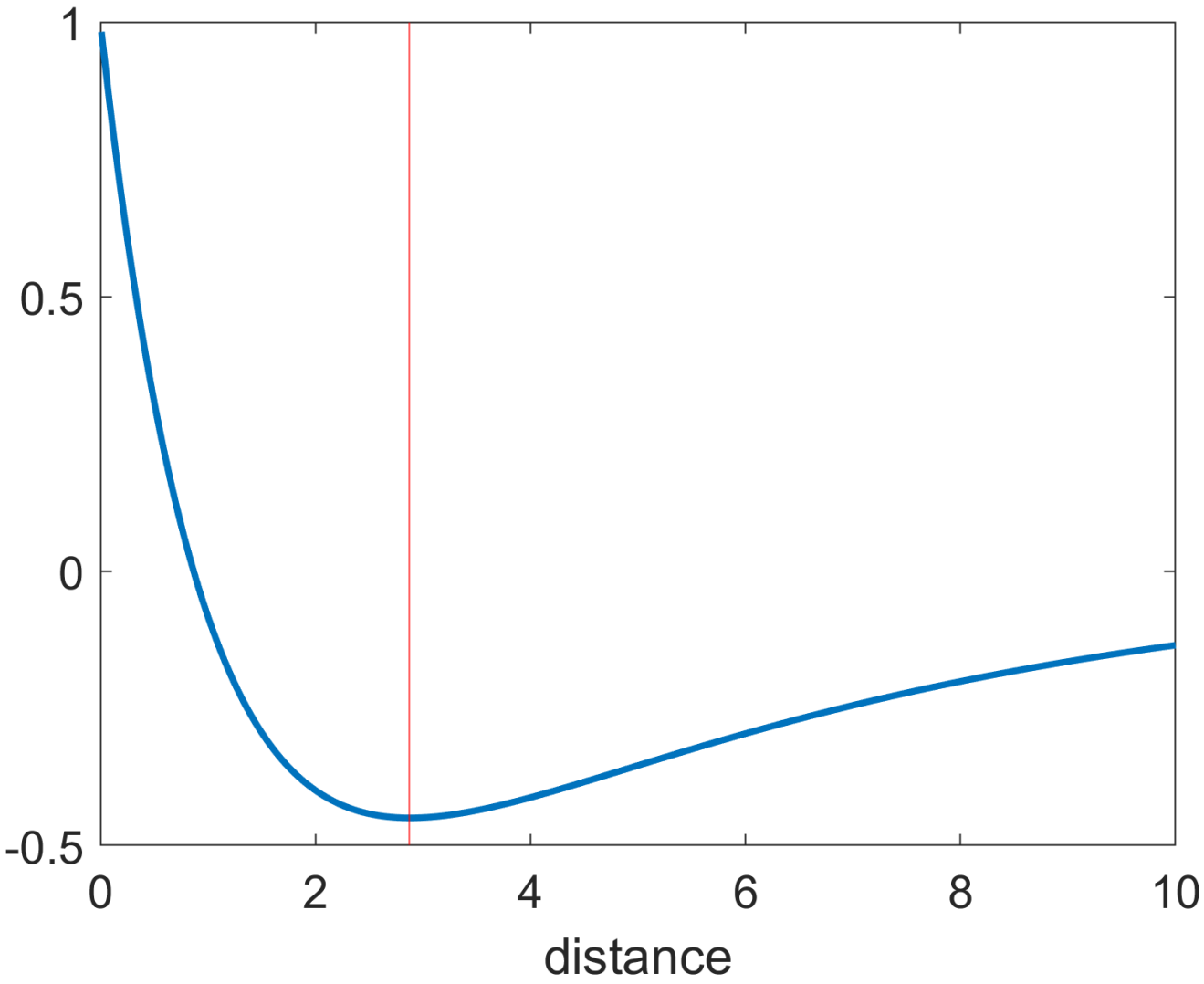
$$f_U(d) = -c_A \exp\left(-\frac{d}{l_A}\right) + c_R \exp\left(-\frac{d}{l_R}\right),$$

  where $c_A \leq 0$ is the constant specifying the strength of the attraction force, $c_R \leq 0$ is the corresponding constant for repulsion force strength and $l_A,\ l_R \leq 0$ are the respective decay rate constants.
- Using f_U we introduce the interaction potential $U_\alpha : \mathbb{X}^n \to \mathbb{X}$ for each agent given by

$$U_\alpha(\boldsymbol{X}(t)) = \sum_{\substack{\beta=1 \\ \beta\neq\alpha}}^{n_a} f_U'(\|x_\alpha - x_\beta\|)\frac{x_\alpha - x_\beta}{\|x_\alpha - x_\beta\|},$$

  where $\|\cdot\|$ refers to the Euclidean distance.
- As long as we choose $c_R > c_A$ and $l_a > l_R$ we have the repulsion from other agents dominating for short distances and the attraction forces dominating for longer distances. The function f_U has then a unique local minimum at an optimal distance where both attraction and repulsion forces are 0.
- This interaction potential is known as **Morse potential**[1].

Visualization of Morse potential between two agents



Movements driven by an Interaction Potential

Task 7:

- Implement an ABM with $n_a = 50$ agents with mobility driven by a Morse potential and visualize the system snapshots. Initialize the agents at random positions within the area $\mathbb{X} = [030] \times [030]$. Use periodic boundary conditions, i.e. identify the positions $(x, 0)$ and $(x, 30)$ as well as $(0, y)$ and $(30, y)$.
- Introduce an additional scaling constant for scaling the distance input of the potential (this way you can control where the optimal distance between agents is). Introduce another scaling constant to scale the total interaction force (this corresponds to increasing/decreasing C_R, C_A with the same factor separately).
- Hint: a good starting choice for parameters is $C_A = 1, C_R = 2, l_A = 5, l_R = 1$.
- How do the dynamics change when you change the parameters? How does the system behave in the case $C_A = 0$?

Task: Combined Movements

Task 8:

- Introduce a scaled Brownian motion in addition to the interaction potential. How do the dynamics change?
- Add an interaction force to the agent system from Task 3. Try some parameter settings including a case where $c_A = 0$. Observe how the interaction force influences the dynamics for different scenarios.

Exponential random variables

Useful properties of exponential random variables:

- Let $\tau_1 \sim \text{Exp}(\lambda_1), ..., \tau_n \sim \text{Exp}(\lambda_n)$ then the minimum $\tau_m := \min\{\tau_1, ..., \tau_n\}$is also exponentially distributed and it holds that

$$\tau_m \sim \text{Exp}\left(\sum_{i=1}^{n} \lambda_i\right)$$

- To realize a random variable $\tau \sim \text{Exp}(\lambda)$ we can draw a uniform random number u and calculate $\log(\frac{1}{\lambda u})$

Next Reaction Method

- To realize a counting process $(\boldsymbol{N}(t))_{t\in\mathbb{T}}$ we draw a sequence of exponential random variables $\tau_1, ..., \tau_n \sim \text{Exp}(\lambda)$ and define iteratively

$$\hat{\boldsymbol{N}}(t+\tau_k) = \hat{\boldsymbol{N}}(t) + 1$$

- Considering a homogeneous compound Poisson processes $(\boldsymbol{X}(t))_{t\in\mathbb{T}}$ with Markov kernel $\mathcal{Q}$ that describes the possible events we draw for each waiting time τ_k also a random variable $Z_k \sim \mathcal{Q}$ to realize an (interaction) event and update

$$\hat{\boldsymbol{X}}(t+\tau_k) = \hat{\boldsymbol{X}}(t) + Z_k.$$

- Using the properties of exponential random variables we can sum the rates of all possible interaction events and draw a waiting time $\tau \sim \text{Exp}(\lambda(\hat{\boldsymbol{X}}(t)))$ that is distributed according to the total adoption rate

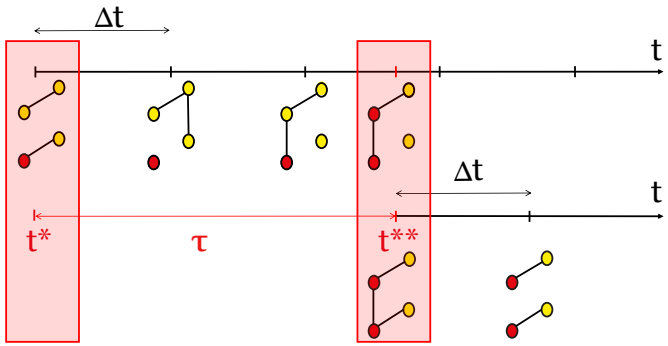
Time-Dependent Jump Intensity

- In our ABMs we have the jump intensity $\lambda(\boldsymbol{X}(t), t)$ also changing between two jump events through the additional dependence on the time t
- We need to compute the numerical integral $\int_t^{t+\tau} \lambda(\boldsymbol{X}(s), s)ds$ such that we can draw a correctly distributed waiting time τ between two events
- We can draw a waiting time $\tau' \sim \text{Exp}(1)$ in advance and solve

$$\tau' = \text{argmin}_\tau \left\{ \int_t^{t+\tau} \lambda(\boldsymbol{X}(s), s)ds \right\}$$

  to calculate a correctly distributed waiting time τ
- This method is also known as (temporal) Gillespie method[2][3]

Event-based ABM simulation



- We combine a first order method for simulating diffusion processes (e.g., Euler-Maruyama scheme) with Gillespie's next reaction method for simulating jump processes.[4]
- The time step size is adaptive and bounded from above by a default time step size Δt for simulation steps without adoption events.

Algorithm 5: Event-based simulation algorithm for ABMs

initialize time $t = 0$ and system state $\boldsymbol{Y}(0) = (\boldsymbol{X}(0), \boldsymbol{S}(0))$ for n_a agents;
choose a time step Δt and time horizon T;
draw $\tau' \sim \text{Exp}(1)$ # exponentially distributed with rate 1;
while $t < T$ **do**
  # calculate rates for adoption events
  for $j = 1 \dots J$ and $\alpha = 1 \dots n_a$ compute the adaption rate function $f_j^{(\alpha)}(\boldsymbol{Y}(t), t)$;
  calculate total adoption rate $\Lambda(t) = \sum_{j=1}^{J} \sum_{\alpha=1}^{n_a} f_j^{(\alpha)}(\boldsymbol{Y}(t), t)$;
  if $\Lambda(t)\ \Delta t > \tau'$ **then**
    # choose adoption event
    define Markov kernel $\mathcal{Q}$ with $\mathcal{Q}(v_j^{(\alpha)}) := \frac{f_j^{(\alpha)}(\boldsymbol{Y}(t), t)}{\Lambda}$;
    draw a state change vector $v_j^{(\alpha)}$ according to $\mathcal{Q}$;
    # status update
    $\boldsymbol{S}\left(t + \frac{\tau'}{\Lambda}\right) = \boldsymbol{S}(t) + v_j^{(\alpha)}$;
    # position update
    draw $\xi \sim \mathcal{N}(0, 1)$;
    $\boldsymbol{X}\left(t + \frac{\tau'}{\Lambda}\right) = \boldsymbol{X}(t) - \nabla V\left(\boldsymbol{Y}(t), t\right) \frac{\tau'}{\Lambda} + \sigma\left(\boldsymbol{Y}(t), t\right) \sqrt{\frac{\tau'}{\Lambda}} \xi$;
    # time update
    $t = t + \frac{\tau'}{\Lambda(t)}$;
    draw new $\tau' \sim \text{Exp}(1)$;
  else
    # position update
    draw $\xi \sim \mathcal{N}(0, 1)^{n_a}$;
    $\boldsymbol{X}\left(t + \Delta t\right) = \boldsymbol{X}(t) - \nabla V\left(\boldsymbol{Y}(t), t\right) \Delta t + \sigma\left(\boldsymbol{Y}(t), t\right) \sqrt{\Delta t} \xi$;
    # time update
    $t = t + \Delta t$;
    $\tau' = \tau' - \Lambda\ \Delta t$;
  end
end
Result: $(\boldsymbol{Y}(t))_{t \leq T}$

Adoption rate functions for spreading dynamics

Adoptions for spreading dynamics:

- We set the adoption rate functions to be $f_{21}^{(\alpha)}(X, S) = 0$ and

$$f_{12}^{(\alpha)}(X, S) = \gamma_{12}\delta_1(s_\alpha) \sum_{\beta=1, \beta\neq\alpha}^{n_a} d_r(x_\alpha, x_\beta)\delta_2(s_\beta)$$

for all agents, with interaction radius $r > 0$ and a rate constant $\gamma_{12} > 0$ for the frequency of interactions. δ_1 and δ_2 denote the discrete indicator functions for status $1/2$ and d_r is the distance indicator function for the interaction radius, i.e. the value is 1 if agents α and β are closer than the interaction radius and 0 else.

Task: Spreading Dynamics (Double Well)

Task 9:

- Expand the ABM from Task 8 to include spreading dynamics. Each agent gets assigned a status variable that can take the values 1 and 2 in addition to the position. Initialize all agents in status 1 and let the system move to an equilibrium for a suitable amount of time (e.g. 10 simulated time units). Choose one agent to change the status to value 2 to initiate the spreading dynamics and start tracking time and snapshots when the spreading starts.
- Adapt your visualization to plot agents with status 1 by blue and agents in status 2 by red dots.
- How do the radius and interaction rate impact the spreading dynamics? How do the mobility dynamics influence the spreading?

Task: Hunter-Gatherer ABM with SI Spreading

Task 10:

- Expand the Hunter gatherer ABM to include an interaction force with $c_A = 0$ (repulsion only) and the spreading dynamics from Task 9 (including a delay period of 4000 simulation years). Adapt the visualization as in Task 9.
- How does the changing landscape impact the spreading? Generate a few realizations with an interaction radius of 5 and an interaction rate of 0.002.
- Advice: With the inclusion of the neighborhood calculations the effort increases at least to $\mathcal{O}(n_a \log(n_a))$. At this point it might make sense to broadcast the progress and visualize on the go while working on the script. Saving the system state before the spreading start and directly initiating the agents with these positions can save you some time while adjusting the model.

Reminder: Opinion Dynamics for the Hunter-Gatherer ABM

Status variable:

- We consider for a feature i a finite set $\mathbb{S}_i$ of $n_s = 10$ traits.
- We assume the trait values to be unordered and that the switch from one arbitrary trait $k \in \mathbb{S}_i$ to any other trait $l \in \mathbb{S}_i$ is possible.
- We choose all rates γ_i to be equal, so we have only one paramater for the rate of first order events.

First order events:

- Spontaneous trait value change for agent α:
  - State change vector: $v_{ikl}^{(\alpha)} := (-k + l)e_i^{(\alpha)}$ encoding the switch from trait k to l in feature i for agent α.
  - Adoption rate function: $g_{ikl}^{(\alpha)}(Y) := \gamma_i \delta_k(s_i)$ with $\gamma_i > 0$ and $k \neq l$.

Second order events:

- Transmission of a trait value to agent α:
  - State change vector: $v_{ikl}^{(\alpha)}$
  - Adoption rate function:

$$f_{ikl}^{(\alpha)}(Y) := \sum_{\substack{\beta=1 \\ \beta \neq \alpha}}^{n} \delta_k(s_i^{(\alpha)}) \delta_l(s_i^{(\beta)}) \cdot A_{\alpha\beta}(Y), \tag{1}$$

Final Task: Hunter-Gatherer ABM with Opinion dynamics

- Adapt the ABM for hunter-gatherers to include Opinon dynamics. Each agent now gets assigned a total of 5 variables, 2 dimensions for position and 3 dimensions (features) for status. Each status can take values between 1 and 10. Choose the rate for first order events to be 0.001 and for interactions to be 0.01. Initiate the agents with random status values and let the status dynamics already run in the delay period before tracking the time.
- Adapt the visualization such that each agent is colored according to an rgb value corresponding to the 3 status variables (this is one reason why we choose 3 features). Use a gray colorscale to visualize the contour lines of the landscape (if you use matlab you can use the one in the data folder)
- Work with shorter simulation/delay periods when working on the code and visualize the system on the go to check whether everything is working as intended. A full run with 10000 simulation years and 4000 additional years to get an equilibrium starting distribution will take half an hour!

References I


Philip M Morse.
Diatomic molecules according to the wave mechanics. ii. vibrational levels.
Physical Review, 34(1):57, 1929.


Daniel T Gillespie.
Exact stochastic simulation of coupled chemical reactions.
The journal of physical chemistry, 81(25):2340–2361, 1977.


Christian L Vestergaard and Mathieu Génois.
Temporal Gillespie algorithm: Fast simulation of contagion processes on time-varying networks.
PLoS computational biology, 11(10):e1004579, 2015.


Nataša Djurdjevac Conrad, Luzie Helfmann, Johannes Zonker, Stefanie Winkelmann, and Christof Schütte.
Human mobility and innovation spreading in ancient times: a stochastic agent-based simulation approach.
EPJ Data Science, 7(1):1–22, 2018.