Pragmatic communicators can overcome asymmetry by exploiting ambiguity

Mark Blokpoel
Radboud University, Donders Institute for Brain, Cognition and Behaviour, The Netherlands

Mark Dingemans, Marieke Woensdregt
Center for Language Studies, Radboud University, The Netherlands

George Kachergis
Department of Psychology, Stanford University, United States

Sara Bögels, Ivan Toni, Iris van Rooij
Radboud University, Donders Institute for Brain, Cognition and Behaviour, The Netherlands

for the CABB team
Language in Interaction Consortium, Radboud University, The Netherlands

Abstract
How can people communicate successfully when they perceive the world differently (asymmetry) and when words can mean different things (ambiguity)? Prior work has appealed to contextual information to blunt the impact of ambiguity, and explicit feedback to resolve misunderstandings caused by asymmetry. We demonstrate that even without contextual scaffolding and before resorting to interactive feedback, communicators using pragmatic inference can actually exploit ambiguity to overcome asymmetry. While intuition suggests that ambiguity only aggravates asymmetry, agent-based simulations show that to pragmatic communicators it is a ‘helping hand’: They can use ambiguity to trade computationally leaner pragmatic inference for costly interactional turbulence. This computational approach to a longstanding linguistic problem establishes a novel theoretical baseline for explaining natural language use.

Keywords: referential communication | pragmatics | rational speech act theory | agent-based modeling

Public language use can smooth over remarkable differences in individual mental representations (Garfinkel, 1967; Kidd, Donnelly, & Christiansen, 2017; Levinson, 2012). This is all the more special because language itself rarely provides explicit and unambiguous encoding of information (Reddy, 1979; Shannon, 1948; Wasow, Perfors, & Beaver, 2005).
Instead, language users have to rely on ostensive-inferential communication using vastly underdetermined signals (Hutchins & Hazlehurst, 1995; Levinson, 2000; Sperber & Wilson, 1996). How can language be both a locus of uncertainty and a tool to resolve it? This was a central conundrum for Locke, who saw language as “the great instrument and common tie of society” and at the same time noted that words “often fail to excite in others (even that use the same language) the same ideas we take them to be signs of” (Locke, 1690).

Prior work has identified two ways to reduce adverse consequences of ambiguity and asymmetry. First, the situation of language use (including environment and prior discourse) provides contextual information that can help us disambiguate (Clark, 1996a; Hockett, 1987): different interpretations attach to “where’s the ball?” depending on whether it is addressed to a friend in full evening dress or a fellow football enthusiast. Second, social interaction offers explicit resources for displaying and calibrating understanding: we can shore up understanding by confirming it or asking for clarification (Clark, 1996b; Dingemanse et al., 2015; Schegloff, 2007). This led to the idea that we can tolerate ambiguity where situational context allows us to disambiguate, and resort to explicit feedback elsewhere (Pickering & Garrod, 2004).

However, open-ended contexts require costly forms of pragmatic inference (Berkum, 2008; Chiappe & Kukla, 1996; van Rooij et al., 2011), and bespoke feedback procedures carry the cost of holding up the interaction, with possible downstream turbulence (Drummond & Hopper, 1991; Skelt, 2012). Explaining how these processes operate in the face of severe time constraints intrinsic to dialogue (Levinson, 2016) is non-trivial: one must identify structures in the situation and interaction that constrain computational demands and minimise interactional hiccups (Schegloff, 2007; van Rooij, 2008; van Rooij, Evans, Müller, Gedge, & Wareham, 2008).

Here, we pursue a middle road by focusing on a computationally leaner form of pragmatic reasoning able to deal with asymmetry between communicators without relying on inference over open-ended situational context or explicit feedback procedures. We adopt the framework of Rational Speech Act theory (RSA) (Frank, Emilsson, Peloquin, Goodman, & Potts, 2017; Frank & Goodman, 2012), in which pragmatic communicators select and interpret referential signals in relation to the relevant subset of signals that could have been used. RSA has been successfully used to explain communication in language games (Frank et al., 2017; Frank & Goodman, 2012; Khani, Goodman, & Liang, 2018), implicatures (Bergen, Levy, & Goodman, 2016; Goodman & Stuhlmüller, 2013), noisy signals (Bergen & Goodman, 2015), convention formation (Cohn-Gordon, Goodman, & Potts, 2018; Hawkins, Frank, & Goodman, 2017), and polite speech (Yoon, Frank, Tessler, & Goodman, 2018). Work in this framework has shown that Bayesian pragmatic agents can
interpret potentially ambiguous signals by probabilistic pragmatic inference (taking into account the other’s pragmatic reasoning, recursively). Although the theory captures both comprehension and production, most work so far has focused on a single agent’s perspective and assumed identical lexicons (but see Hawkins & Goodman, 2017). Here we formalize communicative exchanges between two people across a wide range of ambiguity and asymmetry to investigate under what conditions communication between asymmetric pragmatic interlocutors can remain successful without having to rely on situational or interactional inference.

The findings demonstrate that interlocutors can use ambiguity to counteract the detrimental effects of asymmetry, suggesting that ambiguity in natural language is a desirable design feature, not a defect that has escaped selective pressure (Piantadosi, Tily, & Gibson, 2012). We propose that ambiguity is beneficial because the pragmatic meaning of a signal is not bound by a brittle one-to-many mapping, but is more robustly distributed across the context of things that could have been said. Importantly, this benefit can be attained without engaging in computationally costly reasoning about open-ended contextual information or explicit feedback procedures, suggesting a reason why natural languages have evolved to not just tolerate but actively exploit a moderate degree of ambiguity (Piantadosi et al., 2012; Wasow et al., 2005).

Rational Speech Act model

Of all the things we could say, how do we decide what to say? Suppose someone wants to tell their friend something about a classmate whose name they don’t know. They could mention typical attributes, relations or activities (e.g., ‘the tall one’, ‘the one with red hair’, ‘the teacher’s favorite’ or ‘the gamer’). In its simplest incarnation, the problem is easily solved: if the two people share the same conceptualization of their classmate, have the same vocabulary (no asymmetry), and the signals in their vocabulary uniquely refer to one referent each (no ambiguity), they only need to select the one signal that unequivocally refers to the intended classmate. However, reality is typically less clear-cut and often violates one or more of these conditions. How can people nevertheless reach mutual understanding?

Using cognitive agent-based simulations we investigate how communicative success is affected by asymmetry, ambiguity and the agent’s cognitive ability to perform higher-order pragmatic reasoning. The agents in our simulation produce and interpret signals based on probabilistic pragmatic inference as characterized by the Rational Speech Act (RSA) model (Frank & Goodman, 2012). RSA agents have access to a vocabulary $V = \{s_1, \ldots, s_i\}$, a set of referents $R = \{r_1, \ldots, r_j\}$ and a lexicon $L$ that specifies for each signal to which referents it may refer. Lexicons can reflect weighted relations between signals and referents based on typicality (Graf, Degen, Hawkins, & Goodman, 2016), or frequency of co-occurrence, current dialogue information, or analogies between representations of signals and referents (Blokpoel, Wareham, Haselager, Toni, & Rooij, 2019). However, for simplicity, and without loss of generality, we use binary signal-referent relationships.

In the model, the lexicon $L$ is first transformed into a conditional probability distribution $\delta$, both for a zero-order listener $\delta_{\text{listener}}$ and for a zero-order speaker $\delta_{\text{speaker}}$. Here, $\delta_{\text{listener}}$ specifies the probability distributions over referents $r$ given an (observed) signal $s$, whereas $\delta_{\text{speaker}}$ specifies the probability distributions over signals given an (intended) referent. Zero-order agents are also called literal speakers/listeners (Bergen, Goodman, &
Levy, 2012), since they do not take their interlocutor into account. The $\delta$ are defined by normalizing $L$ over referents or signals respectively:

$$
\delta_{\text{speaker}}(s|r) = \frac{L(s,r)}{\sum_{r' \in R} L(s,r)}
$$

$$
\delta_{\text{listener}}(r|s) = \frac{L(s,r)}{\sum_{s' \in V} L(s',r)}
$$

We can illustrate the RSA model using matrix notation (cf. Frank et al., 2017) and the classroom example from Box 1:

$$
L_{\text{example}} = \begin{bmatrix}
\text{red hair} & \text{Asla} \\
1 & 0 \\
0 & 1 \\
\text{gamer} & 1 \\
\end{bmatrix}
$$

Based on $L_{\text{example}}$, $\delta_{\text{speaker}}$ and $\delta_{\text{listener}}$ can be computed (note the shorthand $\delta_s$ and $\delta_l$):

$$
\delta_s = \begin{bmatrix}
\text{red hair} & \frac{1}{2} & 0 \\
\text{tall one} & 0 & \frac{1}{2} \\
\text{gamer} & \frac{1}{2} & \frac{1}{2} \\
\end{bmatrix}
$$

$$
\delta_l = \begin{bmatrix}
\text{red hair} & 1 & 0 \\
\text{tall one} & 0 & 1 \\
\text{gamer} & \frac{1}{2} & \frac{1}{2} \\
\end{bmatrix}
$$

In this example, ‘gamer’ can refer to both Nora and Asla. This means that to a zero-order listener, the signal ‘gamer’ has two equal probability referents (a uniform distribution). Note that we can already observe why zero-order communication may fail: a zero-order speaker may select with equal probability an ambiguous signal ($\delta_s(\text{red hair}|Nora) = \delta_s(\text{gamer}|Nora)$), with possibility for misinterpretation on the listener’s part ($\delta_l(\text{Nora}|\text{gamer}) = \delta_l(\text{Asla}|\text{gamer})$).

To characterize how speakers $S$ and listeners $L$ perform (pragmatic) audience design, the RSA model introduces $n^{th}$ order conditional probability distributions. Here, we assume that interlocutors assume their partner to be at the same level of pragmatic inference. That is, an $n^{th}$ order speaker assumes to be speaking to an $n^{th}$ order listener and vice-versa. This means that a speaker’s conditional probability distribution is defined analogous to the listener’s, and that an $n^{th}$ order speaker considers the other’s perspective as often as an $n^{th}$ order listener.

The conditional probability distribution for an $n^{th}$ order listener bottoms out in a literal listener and is characterized according to (Frank & Goodman, 2012) (Fig. 1 and Eq. 3–5, left) and the $n^{th}$ order speaker is characterized analogously, bottoming out in a literal speaker (Fig. 1 and Eq. 3–5, right).

---

1Current literature on RSA has not characterized zero order speakers, and assumes that speakers always consider their listener to be 1 order less. In the Supplementary Information we provide a comparison of three flavors of RSA. The main difference is that in the characterization presented here, speakers can be considered to be of slightly higher order than those characterized in (Frank & Goodman, 2012). Since our analysis concerns the contrast between literal and pragmatic agents, this difference does not impact the conclusions of paper.
Figure 1. Rational Speech Act Recursion. The recursive dependencies in the characterization of Rational Speech Act used in this paper. The gray boxes surround the literal listener and speaker, where recursion for speakers and listeners respectively ends. A pragmatic agent (any in the blue or yellow box) takes into account their interlocutor’s perspective (light blue or light yellow box).

\[
\begin{align*}
\Pr_{L_0}^S(r | s) &= \delta_l(r | s) \\
\Pr_{L_n}^S(r | s) &= \frac{\Pr_{S_n}^S(s | r)}{\sum_{r' \in R} \Pr_{S_n}^S(s | r')} \\
\Pr_{S_n}^S(s | r) &= \frac{\Pr_{L_{n-1}}^L(r | s)}{\sum_{s' \in V} \Pr_{L_{n-1}}^L(r | s')} \\
\Pr_{L_n}^L(r | s) &= \frac{\Pr_{S_n}^S(s | r)}{\sum_{r' \in R} \Pr_{S_n}^S(s | r')} \\
\Pr_{S_n}^S(s | r) &= \frac{\Pr_{L_{n-1}}^L(r | s)}{\sum_{s' \in V} \Pr_{L_{n-1}}^L(r | s')}
\end{align*}
\]

The recursion in these distributions, via the parameter \( n \), captures the capacity for speakers and listeners to use higher \((n^{th})\) order pragmatic inferences. Zero order interlocutors base their signals and interpretations purely on the mapping of a signal or referent specified by \( \delta \), not taking the perspective of the other interlocutor into account. For a higher-order speaker \( S_n, n > 0 \), the probability of a signal \( s \) given a referent \( r \in R \) is equal to the probability of that referent given the signal, i.e., the probability that a listener would infer that referent upon hearing the signal. (See Box 1.) We complete the formal characterization of RSA by specifying the computational-level functions for speakers (i.e., RSA PRODUCTION) and listeners (RSA COMPREHENSION):2

\( n^{th} \)-ORDER RSA PRODUCTION

**Input:** A vocabulary \( V \), a set of referents \( R \), a lexicon \( L \), an order of pragmatic inference \( n \) and an intention \( i \in R \).

**Output:** A signal \( s \in V \) that is most probable to lead to \( i \) being inferred

\[
s = \arg\max_{s \in V} \Pr_{S_n}^S(s | i, R).
\]

---

2In this computational-level characterization of RSA, the output is specified by an \( \arg\max_{r \in R} f(r) \) function, selecting the object \( r \) with equal probability from a subset \( R' \subseteq R \) where all \( r \in R' \) maximize \( f(r) \). Alternatively, a softmax \( \max_{r | R} f(r) \) is often used in the literature, where \( r \) is selected with probability proportionate to the value of \( f(r) \).
$n^{th}$-ORDER RSA COMPREHENSION

**Input:** A vocabulary $V$, a set of referents $R$, a lexicon $L$, an order of pragmatic inference $n$ and an observed signal $s \in V$.

**Output:** A referent $r$, where $r = \arg\max_{r \in R} \Pr_L^L(r \mid s)$.

**Agent-based simulation**

This simulation study considered 4,224,000 pairs of agents perform a 1-shot referential communication task (Clark & Wilkes-Gibbs, 1986; Krauss & Weinheimer, 1964) with alternating roles (see Fig. 2). We measure mean communicative success of a pair across all their interactions, where success is operationalized as the intention being the same as the inferred referent $i = r$ (Clark, 1996b).\(^3\)

---

\(^3\)In the main paper, we report a simulation with a vocabulary size of 30, with 8 referents and pairs interacting for $u = 35$ rounds. These particular parameters do not affect the results. See the Supplementary Information (SI) for additional analyses with $|V| = 60$, $|R| = 8$, $|V| = 30$, $|R| = 16$ and $|V| = 8$, $|R| = 8$. 

---

*Figure 2. Simulation. A: Different levels of (uniform) ambiguity for a small example lexicon. B: Different levels of asymmetry between two pairs of lexicons. C: The simulated 1-shot interaction between two agents with alternating roles. We measure mean communicative success $i_u = r_u$ over all rounds $u.$*
We investigate the effect of ambiguity $\alpha$, asymmetry $\gamma$ and order of pragmatic inference $n$ on the agents’ ability to communicate successfully. Signal ambiguity is operationalized as the relative number of referents a signal can refer to, and lexicon ambiguity as mean signal ambiguity. We use $\alpha_i$ to refer to the ambiguity of the lexicon of agent $i$, ranging from $1/|R|$ (no ambiguity) to $|R|/|R|$ (full ambiguity). Asymmetry is operationalized as the relative number of signal-referent relations that are different between two lexicons, ranging from 0 (no asymmetry) to 1 (full asymmetry). The order of pragmatic inference is operationalized as defined by the RSA model. We only investigate up to second order, as previous results show little benefit from going beyond first order (Frank et al., 2017).

Since mean ambiguity $\alpha_i$ can obscure patterns of heavily skewed ambiguity (e.g., mean ambiguity at 0.5 when half of all signals refer to all referents and the other half to none) we assume uniform ambiguity in the main simulation study: each signal refers to the same number of referents ($\text{Var}(\alpha_1) = 0, \text{Var}(\alpha_2) = 0$). This abstraction allows us to investigate precisely the effect of overall ambiguity without it being distorted through potential patterns of skewed ambiguity. In the SI, we relax this assumption and show that the results generalize across two additional simulation studies.

We generate pairs of agents with systematically varying levels of ambiguity and asymmetry. Agent 1’s lexicon $L_1$ is generated for varying degrees of ambiguity $\alpha_1$, under the following constraints: (1) Any referent has at least 1 signal that can uniquely refer to it, (2) each signal gets assigned $\lfloor \alpha_1 |R| - 1 \rfloor$ additional referents at random. To generate Agent 2’s lexicon with varying degrees of overlap one of four options is selected randomly: (1) copy $L_1$ and add $j$ random referents to each signal, (2) copy $L_1$ and remove $j$ random referents from each signal, (3) copy $L_1$ and within a signal swap a random number of signal-referent relations, or (4) generate a lexicon from scratch using the same procedure as for Agent 1.

The result of this generation procedure is a diverse population of agent pairs with lexicons spread across the theoretically possible ambiguity-asymmetry space (see SI). The procedure takes into account a non-trivial relationship between ambiguity and asymmetry that makes it difficult to directly generate randomized pairs of lexicons for any arbitrary combination of ambiguity and asymmetry. In the SI we report the population details and alternative methods for agent generation and find that the results are robust across methods.

Main results

We find three main results. First, we find that in conditions with some asymmetry ($\gamma > 0$) and some ambiguity ($\alpha_1 > \frac{1}{|R|}$ and $\alpha_2 > \frac{1}{|R|}$), a ‘sweet spot’ of moderate ambiguity emerges where pragmatic agents can outperform the best literal agents, even when literal agent pairs have perfectly symmetrical lexicons (Fig. 3). When some ambiguity is present (as in most real-life situations), higher order agents can deal with asymmetry better than literal agents. Pragmatic agents even reach higher communicative success under moderate asymmetry than literal agents do under no asymmetry. The effect holds across a relatively wide range of ambiguity and asymmetry and is robust against variations in agent generation procedure and other auxiliary assumptions (see SI).

Second, while ambiguity and asymmetry are conceptually independent, our formalization shows that they mutually constrain each other. This is easily shown in the limit case of full ambiguity, where no asymmetry is possible. This co-constraint is a mathematical
Figure 3. Main simulation results. A: Mean communicative success for interacting agents with moderately ambiguous lexicons (\(\alpha_1 = \frac{3}{8}, \alpha_2 = \frac{3}{8}\)), where \(\triangle\) indicates peak performance of literal (zero order) agents, \(\triangledown\) the first point where performance of pragmatic (first order) agents drops below that, and the dotted line between them is the additional amount of asymmetry pragmatic agents can tolerate by exploiting ambiguity. B: Increased asymmetry toleration for each possible combination of lexicon ambiguity, showing that for pragmatic agents there is a 'sweet spot' where moderate ambiguity on both sides helps rather than hinders communication. Surrounding smaller panels illustrate communicative success for different combinations of lexicon ambiguity \((\alpha_1, \alpha_2)\).

This relationship constrains the lower and upper bounds of asymmetry, defining the theoretical limits of asymmetry given some level of ambiguity (Fig. 4). A similar relationship also holds for \(\text{Var}(\alpha_i) > 0\), as can be observed in the simulation studies reported in SI. Natural language lexicons will, by mathematical necessity, be similarly constrained.

Third, we replicate and generalize a key finding of rational speech act theory: higher orders of pragmatic inference have diminishing returns (Frank et al., 2017). In particular, while first order agents mostly outrank zero order agents in terms of communicative suc-
Figure 4. **Ambiguity and asymmetry are co-constrained.** A boxplot of the theoretical range of asymmetry ($\gamma_{\min}, \gamma_{\text{mean}}, \gamma_{\max}$) as defined by Eq. 6–8 for $|R| = 8$. 

cess, second order agents are almost indistinguishable from first order agents. Because we systematically manipulate ambiguity and asymmetry, we can now also observe that this effect holds across all degrees of ambiguity and asymmetry (Fig. 5, right panel).

Figure 5. **Replication and generalization results.** Colors indicate combinations of ambiguity. **Left:** Mean success of zero order agents is subtracted from first order agents. First order agents achieve higher communicative success, except when there is no ambiguity $\alpha_1 = \alpha_2 = 1/8$ or asymmetry $\gamma > 0.5$. **Right:** Mean success of first order agents is subtracted from second order agents. There is no systematic benefit going beyond first order across all combinations of ambiguity and asymmetry.
Discussion

How can people understand each other even when they represent the world differently? Interacting Rational Speech Act agents show that pragmatic communicators can overcome asymmetry by exploiting the ambiguity in their lexicons. This proves that there exist computationally leaner mechanisms that allow interlocutors to counter the detrimental effects of asymmetry prior to resorting to more costly forms of pragmatic inference and interactive repair. Our results establish new theoretical baselines for pragmatic communication under naturalistic conditions of asymmetries in lexical knowledge.

Our finding that pragmatic agents can garner a functional benefit from a moderate degree of lexical ambiguity has consequences both for explaining human interaction and for the cultural evolution of natural language lexicons. Work on natural language use suggests that ambiguity is the rule rather than the exception (Wasow et al., 2005), and that communicative interaction can often proceed despite differences in conceptual representations (Enfield, 2007; Garfinkel, 1967). While interactive repair occurs every few minutes in conversation (Dingemanse et al., 2015), there is evidence that participants often let communicative trouble pass to avoid halting the interaction (Jefferson, 1987, 2018), with one study estimating this may happen in up to 25% of cases of possible trouble (Caissie & Gibson, 1997). The results help explain the high tolerance of ambiguity in interaction by providing an underlying mechanism by which it can be exploited.

The results also have relevance for the cultural evolution of natural language lexicons. Culturally evolving lexicons are shaped by competing pressures for representational efficiency and communicative utility (Brochhagen, Franke, & van Rooij, 2018; Kirby, Tamrize, Cornish, & Smith, 2015). While ambiguity has been seen as a sign of non-optimal design (Chomsky, 2002) or at most a by-product of the pressure for efficient encoding (Plantadosi et al., 2012), the results imply that it can provide a direct functional communicative benefit in coping with asymmetry. Under the right conditions, the ever-present challenge of knowledge asymmetry can at least partially be alleviated by lexicons with moderate ambiguity by enabling greater communicative success in a wider range of situations. In cultural evolutionary terms, the ‘ambiguity sweet spot’ we have identified may over time favor lexicons with a moderate degree of ambiguity. This predicts different evolutionary trajectories for populations of pragmatic agents with symmetric and asymmetric lexicons (see Fig. 6; Barr, 2004; Brighton, Kirby, & Smith, 2003; Thompson, Kirby, & Smith, 2016).

Our formal results provide a window into the theoretical space of ambiguity and asymmetry in relation to communicative success. They also bring new questions into view. First, a key move in Rational Speech Act theory, adopted in our simulations too, is to restrict the domain of pragmatic inference to the relevant subset of all words and referents in principle available to the agent. Psycholinguistic evidence suggests that people can indeed rapidly narrow down to relevant sets of vocabulary and referents (Levitt, 1989; Sperber & Wilson, 1996), but how this is done computationally remains a vexing question not just for RSA models, but for computational cognitive science at large (Chiappe & Kukla, 1996; Fodor, 2000).

The present findings also qualify a second long-standing question on the nature of the relationship between signal and referent. While much work assumes that people entertain

---

4See the SI for a proof that RSA is computationally tractable, and why repair is more costly.
identical conceptual representations (Hasson, Ghazanfar, Galantucci, Garrod, & Keysers, 2012; Pickering & Garrod, 2013), full conceptual alignment is not necessary for referential success (Enfield, 2007; Garfinkel, 1967). We suggest that signal-referent relations are best seen as graded analogical relationships between construals of signal and referent (Blokpoel, Wareham, Haselager, Toni, & Rooij, 2018; Gentner & Markman, 1997), where multiple construals are possible and people may entertain different construals privately. On this view, our simulations suggest how a moderate amount of ambiguity coupled with asymmetry brings a larger chunk of the lexicon within the scope of pragmatic reasoning towards analogical construals, and therefore allows communicative success.

Explaining human interaction requires identifying cognitively opaque mechanisms that are not easily reverse engineered (Tenenbaum, Kemp, Griffiths, & Goodman, 2011). The relations between ambiguity and asymmetry uncovered here illustrate the necessity to use a formal approach for resolving those mechanisms. The helping hand of ambiguity may well be obscured in controlled experimental settings where more overt mechanisms such as situational disambiguation or explicit feedback take precedence, yet it may constitute a substantial part of explaining human communication. In an interactive and dynamic system like human language, simulating the interactional consequences of postulated theories and linking them to empirical findings about interaction can lead us to discover underlying mechanisms.

To conclude, our simulation study has laid bare an otherwise invisible principle. Pragmatic communicators can exploit the ambiguity of their lexicons to overcome asymmetries in mental representations, even before taking recourse to computationally costlier solutions like contextual scaffolding or interactive feedback. This allows them to trade computationally lean pragmatic inference for costly interactional turbulence. These results go beyond prior information-theoretic explanations for the occurrence of ambiguity, unify scattered observations on the pervasiveness of ambiguity in interaction, and establish important new theoretical baselines for explaining communicative success in naturalistic settings.

Figure 6. Predicted evolution of ambiguity: A: In a population of asymmetrical pragmatic agents, asymmetry is a pressure on communicative success. Lexicons evolve over time towards a sweet spot of moderate ambiguity. Different levels of ambiguity at origin have different evolutionary trajectories. B: In a population of symmetrical agents, ambiguity has no functional benefit. Lexicons evolve towards low ambiguity, unless other pressures are present (e.g., representational efficiency Piantadosi, Tily, & Gibson, 2012).
Acknowledgements

MB is supported by Netherlands Organization for Scientific Research (NWO) (Gravitation Grant 024.001.006 of the Language in Interaction consortium, LiI). MD is supported by NWO (016.Vidi.185.205). IT is supported by NWO (453-08-002 and Gravitation Grant 024.001.006 of the LiI). We thank Ron Hommelsheim for his technical support.

References


**Box 1:** Illustration of the Rational Speech Act model.

We illustrate how a first-order listener computes a probabilistic lexicon and infers the referent for a given signal. Based on Eq. 3–5 the recursion can be described as follows (abstracting away the normalization factors):

\[
\text{Pr}_{L_1} \propto \text{Pr}_{S_1} \propto \text{Pr}_{L_0} = \delta_l
\]

Representing the conditional probability of referents given signals (and vice versa) as a matrix, we start at the bottom of the recursion, working up:

<table>
<thead>
<tr>
<th></th>
<th>Nora</th>
<th>Asla</th>
</tr>
</thead>
<tbody>
<tr>
<td>red hair</td>
<td>$1/1\frac{1}{2}$</td>
<td>0</td>
</tr>
<tr>
<td>tall one</td>
<td>0</td>
<td>$1/1\frac{1}{2}$</td>
</tr>
<tr>
<td>gamer</td>
<td>$1/2/1\frac{1}{2}$</td>
<td>$1/2/1\frac{1}{2}$</td>
</tr>
</tbody>
</table>

\[
\text{Pr}_{L_0} = \delta_l =
\begin{bmatrix}
  1 & 0 \\
  0 & 1 \\
  \frac{1}{2} & \frac{1}{2}
\end{bmatrix}
\]

To compute \(\text{Pr}_{S_1}\) we divide each \(\text{Pr}_{L_0}(r | s)\) by the normalization factor (the sum of probabilities for a referent given all possible signals, $1\frac{1}{2}$ for Nora and $1\frac{1}{2}$ for Asla):

<table>
<thead>
<tr>
<th></th>
<th>Nora</th>
<th>Asla</th>
</tr>
</thead>
<tbody>
<tr>
<td>red hair</td>
<td>$2/3$</td>
<td>0</td>
</tr>
<tr>
<td>tall one</td>
<td>0</td>
<td>$2/3$</td>
</tr>
<tr>
<td>gamer</td>
<td>$1/3/2/3$</td>
<td>$1/3/2/3$</td>
</tr>
</tbody>
</table>

\[
\text{Pr}_{S_1} =
\begin{bmatrix}
  1/1\frac{1}{2} & 0 \\
  0 & 1/1\frac{1}{2} \\
  1/2/1\frac{1}{2} & 1/2/1\frac{1}{2}
\end{bmatrix} =
\begin{bmatrix}
  2/3 & 0 \\
  0 & 2/3 \\
  1/3 & 1/3
\end{bmatrix}
\]

Each step in the recursion normalizes the distribution over either the referents or signals (Frank, Emilsson, Peloquin, Goodman, & Potts, 2017). Finally, we compute \(\text{Pr}_{L_1}\) using normalization factors $2/3$ for ‘red hair’, $2/3$ for ‘tall one’ and $2/3$ for ‘gamer’:

<table>
<thead>
<tr>
<th></th>
<th>Nora</th>
<th>Asla</th>
</tr>
</thead>
<tbody>
<tr>
<td>red hair</td>
<td>$2/3/2/3$</td>
<td>0</td>
</tr>
<tr>
<td>tall one</td>
<td>0</td>
<td>$2/3/2/3$</td>
</tr>
<tr>
<td>gamer</td>
<td>$1/3/2/3$</td>
<td>$1/3/2/3$</td>
</tr>
</tbody>
</table>

\[
\text{Pr}_{L_1} =
\begin{bmatrix}
  1/1\frac{1}{2} & 0 \\
  0 & 1/1\frac{1}{2} \\
  1/2/1\frac{1}{2} & 1/2/1\frac{1}{2}
\end{bmatrix} =
\begin{bmatrix}
  2/3 & 0 \\
  0 & 2/3 \\
  1/3 & 1/3
\end{bmatrix}
\]

Observe that in this particular example \(\text{Pr}_{L_1} = \delta_s\), but this need not always be the case. However, it is one particular example of why (in certain cases) there is no additional benefit of going through higher levels of pragmatic reasoning (Frank, Emilsson, Peloquin, Goodman, & Potts, 2017).

The listener inference, as defined in RSA **INTERPRETATION**, can now be computed by multiplying \(\text{Pr}_{L_1}\) with a probabilistic representation of a signal (e.g. ‘red hair’):

\[
\text{Pr}^R_{L_1}(R | \text{red hair}) = \begin{bmatrix}
  \frac{2}{3} & \frac{2}{3} & 0 \\
  0 & \frac{2}{3} & \frac{2}{3} \\
  \frac{1}{3} & \frac{2}{3} & \frac{2}{3}
\end{bmatrix} \cdot
\begin{bmatrix}
  \frac{2}{3} & 0 \\
  0 & \frac{2}{3} \\
  \frac{1}{3} & \frac{2}{3}
\end{bmatrix} =
\begin{bmatrix}
  2/3 & 0
\end{bmatrix}
\]

Here, upon hearing ‘red hair’, a first order listener would infer the most probable referent: Nora.
Supplementary Information for Pragmatic communicators can overcome asymmetry by exploiting ambiguity

Mark Blokpoel
Radboud University, Donders Institute for Brain, Cognition and Behaviour, The Netherlands

Mark Dingemanse
Center for Language Studies, Radboud University, The Netherlands

George Kachergis
Department of Psychology, Stanford University, United States

Sara Bögels, Ivan Toni, Iris van Rooij
Radboud University, Donders Institute for Brain, Cognition and Behaviour, The Netherlands

for the CABB team
Language in Interaction Consortium, Radboud University, The Netherlands
In this supplementary information document we report additional analyses of the main simulation study, a robustness study with additional supplementary simulations, information and documentation regarding the simulation software and data.

**Different characterizations of Rational Speech Act theory**

Here we detail three different characterizations of rational speech act theory, the original by Frank and Goodman (2012), the game-theoretic variant by Franke and Degen (2016), and the variant used in the main paper. The main difference of the latter is that it assumes a speaker’s conditional probability distribution is defined analogous to the listener’s, and that an \( n \)th order speaker considers the other’s perspective as often as an \( n \)th order listener. For the mathematical definitions we refer the reader to the respective literature, but a graphical summary of the recursion can be found in Fig. 1. Table 1 provides examples for each theory, where one can observe that compared to Frank and Goodman (2012) the characterization from the main paper assumed speakers to be of slightly higher order of reasoning.

**Additional analyses**

Fig. 2 illustrates the population distribution of the agent pairs generated in the main simulation. While there is good coverage of the theoretical possible range of agents, it can be seen that there are gaps in this range. Furthermore, the main simulation assumed uniform ambiguity which possibly is an unrealistic assumption and potentially a cause of the main effect reported. In the next section, we demonstrate that the findings are robust.
Table 1

**RSA variants.** The different characterizations and their recursive formalizations. The black lines denote the recursion as defined. See Eq. 1–5 in the main paper for the characterization on the right.

<table>
<thead>
<tr>
<th>RSA variant</th>
<th>( L_0^L )</th>
<th>( L_1^L )</th>
<th>( S_0^S )</th>
<th>( L_1^S )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frank and Goodman (2012)</td>
<td>( R(\mathcal{L}) )</td>
<td>( R(C(R(\mathcal{L}))) )</td>
<td>undefined</td>
<td>( C(R(\mathcal{L})) )</td>
</tr>
<tr>
<td>Franke and Degen (2016)</td>
<td>( R(\mathcal{L}) )</td>
<td>( R(C(\mathcal{L})) )</td>
<td>( C(\mathcal{L}) )</td>
<td>( C(R(\mathcal{L})) )</td>
</tr>
<tr>
<td>main paper</td>
<td>( R(\mathcal{L}) )</td>
<td>( R(C(R(\mathcal{L}))) )</td>
<td>( C(\mathcal{L}) )</td>
<td>( C(R(C(\mathcal{L}))) )</td>
</tr>
</tbody>
</table>

across different agent generation procedures, agents distribution, and do not rely on an uniform ambiguity assumption.

**Figure 2. Main simulation agent distribution:** The distribution is a log-transformation of the number of agent pairs per condition.

**Supplementary simulations: Investigating robustness**

We present two alternative pair generation procedures that break the uniform ambiguity assumption and change the distribution of agent pairs. The results of these supplementary simulations show that the effects observed in the main simulation study are robust with respect to particular agent generation procedures (Fig. 3).
Pair generation procedure I

Agent 1’s lexicon is generated given a random density parameter $0 < d \leq 1$. Each signal-referent relationship in this lexicon is true with $\Pr(d)$. Agent 2’s lexicon is generated given a random mutation parameter $0 \leq m \leq 1$ and Agent 1’s lexicon. Each signal-referent relationship has $\Pr(m)$ of being different. In total, 585,219 pairs were generated (Panel B, Fig. 4 for their distribution).

Pair generation procedure II

To generate Agent 1’s lexicon, this generation procedure generates randomized binary strings of length 6 for all signals and referents. Then, based on randomized threshold parameter, a signal-referent relationship is true if the hamming-distance between a the representation of a signal and referent exceeds the threshold. Agent 2’s lexicon is generated by copying and randomly modifying the string representations of the referents. Then a lexicon is constructed according to the same threshold and hamming-distance. In total, 1,330,452 pairs were generated (Panel D, Fig. 4 for their distribution).

Auxiliary assumptions

Both generation procedures lead to the auxiliary assumption of varying ambiguity (Panels A and C, Fig. 4). Within any given lexicon, signals can refer to any number of referents. This potentially skews results by allowing for a lexicon to have a low mean ambiguity, but contain several very high ambiguity and low ambiguity signals. We observe, however, that most pairs have a variance in ambiguity around 1, suggesting most lexicons are not skewed too extremely.

Results

In both supplementary simulation studies we observe the same three-way interaction (Fig. 3) and replication results (Fig. 5) reported in the main article, providing converging evidence that the main results are robust and not due to the auxiliary assumptions. We note that an increased capacity of first order agents for handling asymmetry can also be observed for $\alpha_1 = 1$ or $\alpha_2 = 1$. However, this is due to the fact that these conditions may contain lexicons that on average have 1-to-1 mappings, but in fact contain ambiguity in parts of their lexicon which can be exploited by pragmatic agents. This is also the reason we observe that, contrary to the main study, the effect is also present when $\alpha_1 \neq \alpha_2$, as long as $\alpha_i > 1$.

Differently sized lexicons

We ran additional simulations with uniform ambiguity and differently sized lexicons, specifically: $|V| = 60$, $|C| = 8$; $|V| = 8$, $|C| = 8$; and $|V| = 30$, $|C| = 16$. These simulations also all show the same three-way interaction between asymmetry, ambiguity and order of pragmatic inference on communicative success. See Figure 6.
Computational complexity of RSA compared to interactional repair

We provide a ‘back of the envelope’ proof concept that RSA as characterized in the main paper is computationally lean (i.e., cubic $O(n^3)$) and we show that more interactional repair is computationally more complex.

**RSA Production and RSA Comprehension are tractable**

Let $m$ be $\max(|V|, |R|)$ and let $o$ be the order of pragmatic inference. We can compute either $o^{th}$ order production or comprehension in three stages:

- Compute $o^{th}$ order lexicon
- Compute posterior distribution given intention or signal
- Find argmax in posterior distribution

Stage 1 is defined by Formulas 1–6 and is implemented by normalizing across signals and referents. For production, a $n^{th}$ order speaker has to normalize $2n^2$ times, where each normalization takes $2i^2$ steps (first summarizing rows or columns takes $i^2$ steps, then dividing each cell by the sum also takes $i^2$ steps). Hence, production takes $4n^2i^2$ steps. Comprehension requires one addition recursive step for a total of $(4n + 1)i^2$ steps. Assuming the input size $n = \max(m, o)$ computational complexity of stage 1 is upper bounded by $(4n + 1)n^3 = O(n^3)$.

Stage 2 is defined as the dot product between the $o^{th}$ order lexicon and a probability distribution over referents or the vocabulary. Computing the dot product is of quadratic complexity, stage two is $O(n^2)$.

Stage 3 is implemented by a simple search over the posterior distribution, linear in time complexity $O(n)$. The combined complexity of our implementation is $O(n^3) + O(n^2) + O(n) = O(n^3)$, hence RSA is polynomial time computable. While cubic time complexity is not extremely efficient (compared to (sub-)linear), it is considered computationally tractable (Garey & Johnson, 1979).

A caveat here is necessary, namely that any extension to the version of RSA as presented in this paper may render it intractable. For example, extending RSA to capture people’s ability to communicate multiple intentions may substantially change its computational demands (van Rooij et al., 2011). Regardless of RSA’s (in)tractability, it is in any case resource leaner than interactional repair, as repair not only contains RSA as a subcomputation, but also requires communicators to invest interactional resources.

**Interactional repair is computationally difficult**

Repair is minimally harder than pragmatic inference as characterized by RSA because any form of repair will have pragmatic inference as a subprocess. However, the extent of the computational demands of repair becomes more visible when one considers that repair involves updating one’s lexicon to based on interactional history. This is no trivial task, as given any piece of evidence, one may have to consider all possible alternative lexicons. For any given binary lexicon, there exist $2^{|V| \times |R|}$ alternatives. Even for a small lexicon, say $|V| = 6$ and $|R| = 4$, this leads to $2^{6 \times 4} - 1 = 2^{24} - 1 = 16,777,216 - 1$ alternatives and
this number grows exponentially when the size of the lexicon grows (see Table 2). Even with a graded approach, i.e., a lexicon with values between 0 and 1, one would have to be able to account for people’s sudden jumps in lexicon updating, i.e., situations where people only require a single datapoint to infer the correct lexicon update. This potentially requires a graded updating approach augmented with some model-based approach, where the model-based approach will potentially need to consider all possible alternative lexicons.

Table 2
The number of alternative lexicons for differently sized lexicons.

<table>
<thead>
<tr>
<th>Lexicon size</th>
<th>Number of alternatives</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>$2^4 - 1 = 15$</td>
</tr>
<tr>
<td>10</td>
<td>$2^{10} - 1 = 1023$</td>
</tr>
<tr>
<td>20</td>
<td>$2^{20} - 1 = 1048574$</td>
</tr>
<tr>
<td>50</td>
<td>$2^{50} - 1 = 1125899906842623$</td>
</tr>
<tr>
<td>75</td>
<td>$2^{75} - 1 = 37778931862957161709567$</td>
</tr>
<tr>
<td>100</td>
<td>$2^{100} - 1 = 1267650600228229401496703205375$</td>
</tr>
</tbody>
</table>

Simulation software and data

The simulation software, a user guide, tutorials, data and analysis scripts can be found at our GitHub repository: [link]. Table 3 provides an overview of all available resources of this project.

Software guide

We envision several possible use-cases for the software. Below are our recommendations for each use-case.

- **Analysis**: Download (1) and (2), verify or modify (2) using R Studio. Alternatively, one can generate their own data by running the simulations.

- **Understanding RSA implementation**: Download (5), follow installation instructions at (4) and follow the tutorial.

- **Understanding simulation experiments**: Download (6), follow installation instructions at (4) and follow the tutorial.

- **Run full simulation**: Simulations can be run via (7–9), via command line or by manually compiling the code. Follow instructions at (4).

- **Adapt simulations**: First follow both tutorials at (5) and (6); follow the advanced instructions at (4) to setup your development copy and finally, using the software documentation (9) and (10) and this paper for reference, implement your adaptations.

1 Please note that the ‘running full simulation’ and ‘adapt simulations’ use-cases will require a Linux or OSX operating system.

2 See footnote[1]
### Table 3

An overview of all available resources. To run a Scala notebooks on your own machine, follow these instructions.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data sets</td>
<td></td>
</tr>
<tr>
<td>a. Main simulation data: uniform_30-8.zip</td>
<td></td>
</tr>
<tr>
<td>b. Supplementary simulation, procedure I:</td>
<td>randomized_30-8.zip</td>
</tr>
<tr>
<td>c. Supplementary simulation, procedure II:</td>
<td>binstrings_30-8.zip</td>
</tr>
<tr>
<td>d. Supplementary simulation,</td>
<td>uniform_8-8.zip</td>
</tr>
<tr>
<td>e. Supplementary simulation,</td>
<td>uniform_60-8.zip</td>
</tr>
<tr>
<td>Analysis script</td>
<td></td>
</tr>
<tr>
<td>2. R-notebook (.Rmd)</td>
<td></td>
</tr>
<tr>
<td>3. R-notebook preview (.html)</td>
<td></td>
</tr>
<tr>
<td>Simulation software</td>
<td></td>
</tr>
<tr>
<td>4. GitHub repository (w. guide)</td>
<td></td>
</tr>
<tr>
<td>Interactive simulation tutorial</td>
<td></td>
</tr>
<tr>
<td>part 1: RSA</td>
<td></td>
</tr>
<tr>
<td>5. Scala notebook (.ipynb)</td>
<td></td>
</tr>
<tr>
<td>Interactive simulation tutorial</td>
<td></td>
</tr>
<tr>
<td>part 2: Simulation</td>
<td></td>
</tr>
<tr>
<td>6. Scala notebook (.ipynb)</td>
<td></td>
</tr>
<tr>
<td>Simulation experiments</td>
<td></td>
</tr>
<tr>
<td>7. Uniform simulation experiment (.ipynb)</td>
<td></td>
</tr>
<tr>
<td>8. Random simulation experiment (.ipynb)</td>
<td></td>
</tr>
<tr>
<td>9. Structured simulation experiment (.ipynb)</td>
<td></td>
</tr>
<tr>
<td>Lanag Core documentation</td>
<td></td>
</tr>
<tr>
<td>10. API Scaladoc</td>
<td></td>
</tr>
<tr>
<td>Lanag Ambiguity Helps documentation</td>
<td></td>
</tr>
<tr>
<td>11. API Scaladoc</td>
<td></td>
</tr>
</tbody>
</table>

Questions, bugs and feedback can be reported as issues in the GitHub repository.

**Reproduction and replication**

We provide access to the data generated by the simulation experiment as run by the authors through GitHub (Table 3). This dataset—in combination with the simulation software, tutorials and analysis scripts—is intended for reproduction purposes. However, we note that reproduction studies for simulation study only serve the purpose of validating the analyses presented in the main paper (cf. Rougier et al., 2017). Replication, in contrast, would require re-implementation of our software according to the theory specified in the main paper (cf. Cooper & Guest, 2014). A third, middle-ground option is to extend / adapt our simulation software to implement robustness tests as we have reported here in the Supplementary Information. This can vary from simply changing parameters and re-running the simulation study to implementing different agent generation procedures.

**Simulation software**

The simulation software is published in two parts. A generic framework (Lanag Core) and the implementation of the simulation studies presented in this paper (Lanag Ambiguity
Table 4 provides an overview of the Scala classes and which parts of the theory they implement. Those wishing to extend or adapt the software may pick and choose which classes to re-implement, extend or adapt. As long as the application programming interface (API) is respected, the simulations as defined in the main paper and supplementary information should run fine. For full documentation of these classes see the projects Scaladoc and two tutorials are provided to provide insight into the implementation of RSA (Table 3).

Running the simulation software, there is a configuration file `application.conf` that can be used to change several settings (see Table 5), such as number of samples, vocabulary and context sizes, etc.

Table 4
An overview of the simulation implementation. Packages are located in the main package `com.markblokpoel.lanag`.

<table>
<thead>
<tr>
<th>Scala class</th>
<th>Package</th>
<th>Theory part implemented by class</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA1ShotAgent</td>
<td>ambiguityhelps</td>
<td>n/a (generic RSA agent class)</td>
</tr>
<tr>
<td>RSA1ShotSpeaker</td>
<td>ambiguityhelps</td>
<td>$n^\text{th}$-order RSA Production</td>
</tr>
<tr>
<td>RSA1ShotListener</td>
<td>ambiguityhelps</td>
<td>$n^\text{th}$-order RSA Interpretation</td>
</tr>
<tr>
<td>RSA1ShotInteraction</td>
<td>ambiguityhelps</td>
<td>Simulation setup (Main article, Fig. 1)</td>
</tr>
<tr>
<td>Lexicon</td>
<td>core</td>
<td>Rational Speech Act theory (Main article, Equations 1–6)</td>
</tr>
<tr>
<td>UniformPairGenerator</td>
<td>ambiguityhelps</td>
<td>Main simulation: Pair generation procedure</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RandomPairGenerator</td>
<td>ambiguityhelps</td>
<td>Supplementary simulation: Pair generation procedure I</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>StructuredPairGenerator</td>
<td>ambiguityhelps</td>
<td>Supplementary simulation: Pair generation procedure II</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UniformExperiment</td>
<td>ambiguityhelps</td>
<td>Starts main simulation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RandomExperiment</td>
<td>ambiguityhelps</td>
<td>Starts supplementary simulation w. Pair generation procedure I</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>StructuredExperiment</td>
<td>ambiguityhelps</td>
<td>Starts supplementary simulation w. Pair generation procedure II</td>
</tr>
</tbody>
</table>

References
Figure 3. **Additional simulation results.** Panels A and B are the results of an additional simulation using pair generation procedure I (randomized generation). Panels C and D are the results of an additional simulation using pair generation procedure II (binary strings generation). A and C: Mean communicative success for interacting agents with moderately ambiguous lexicons ($\alpha_1 = 3$, $\alpha_2 = 3$), where $\triangle$ indicates peak performance of literal agents, $\triangledown$ the first point where performance of pragmatic agents drops below that, and the dotted line between them is the additional amount of asymmetry pragmatic agents can tolerate by exploiting ambiguity. B and D: Increased asymmetry toleration for each possible combination of lexicon ambiguity, showing the same effect as reported in the main paper, viz. that for pragmatic agents there is a ‘sweet spot’ where moderate ambiguity on both sides helps rather than hinders communication.
Figure 4. Additional simulation distributions. Panels A and B are the results of an additional simulation using pair generation procedure I (randomized generation). Panels C and D are the results of an additional simulation using pair generation procedure II (binary strings generation). A and C: The variance in lexicon ambiguity (log-scale) across all agent pairs. Ambiguity for both agents is collapsed. B and D: The distribution is a log-transformation of the number of agent pairs per condition.

Figure 5. **Additional simulation replication and extension.** Panel A is the results of an additional simulation using pair generation procedure I (randomized generation). Panel B is the results of an additional simulation using pair generation procedure II (binary strings generation). Colors indicate combinations of ambiguity. **Within the panels left:** Mean success of zero order agents is subtracted from first order agents. First order agents achieve higher communicative success, except when there is no ambiguity $\alpha_1 = \alpha_2 = 1/8$ or asymmetry $\gamma > 0.5$. **Within the panels right:** Mean success of first order agents is subtracted from second order agents. There is no benefit going beyond first order inference across all combinations of ambiguity and asymmetry.
Table 5
An overview of the simulation parameters and settings. Can be set in file application.conf.

<table>
<thead>
<tr>
<th>Simulation</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main simulation</td>
<td>vocabulary-size</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>context-size</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>sample-size</td>
<td>2000</td>
</tr>
<tr>
<td></td>
<td>interaction-length</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>beta</td>
<td>inf</td>
</tr>
<tr>
<td></td>
<td>random-seed</td>
<td>9044</td>
</tr>
<tr>
<td></td>
<td>consistent.change-resolution</td>
<td>0.1</td>
</tr>
<tr>
<td>Supplementary I</td>
<td>vocabulary-size</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>context-size</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>sample-size</td>
<td>2000</td>
</tr>
<tr>
<td></td>
<td>interaction-length</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>beta</td>
<td>inf</td>
</tr>
<tr>
<td></td>
<td>random-seed</td>
<td>9044</td>
</tr>
<tr>
<td></td>
<td>random.density-resolution</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>random.mutation-resolution</td>
<td>0.1</td>
</tr>
<tr>
<td>Supplementary II</td>
<td>vocabulary-size</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>context-size</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>sample-size</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>interaction-length</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>beta</td>
<td>inf</td>
</tr>
<tr>
<td></td>
<td>random-seed</td>
<td>9044</td>
</tr>
<tr>
<td></td>
<td>structured.representation-length</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>structured.threshold-resolution</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td>structured.change-resolution</td>
<td>0.05</td>
</tr>
<tr>
<td>Different Lexicons</td>
<td>vocabulary-size</td>
<td>60, 8, 30</td>
</tr>
<tr>
<td></td>
<td>context-size</td>
<td>8, 8, 16</td>
</tr>
<tr>
<td></td>
<td>sample-size</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>interaction-length</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>beta</td>
<td>inf</td>
</tr>
<tr>
<td></td>
<td>random-seed</td>
<td>9044</td>
</tr>
<tr>
<td></td>
<td>consistent.change-resolution</td>
<td>0.1</td>
</tr>
</tbody>
</table>
All additional simulations show the same main effect as reported in the main paper, suggesting that the main effect is not due to a particular lexicon size (or ratio between \(V\) and \(R\)). Panels A and B are the results of an additional simulation with lexicons sized \(|R| = 8, |V| = 8\). Panels C and D are the results of an additional simulation with lexicons sized \(|R| = 8, |V| = 60\). Panels E and F are the results of an additional simulation with lexicons sized \(|R| = 16, |V| = 30\).