Simulations in Pseudocode. We consider the following set of input parameters:

- n: the number of agents;
- Δp: the margin by which the agents are on average more likely to include p^* in their belief;
- P_{min}: the minimal success probability required for VfB to identify the correct bin;
- t: the percentage of the maximal bins constructed;
- α: the weight balancing truth and informativeness when computing the epistemic value;
- l: the number of times imprecise beliefs are constructed for one parameter input.

Moreover, we restate the bound provided on the number of bins and refer to its outcome for a specific set of input parameters as *max*:

Theorem 1. *For n independent agents where $\Delta p \in (0,1)$, the worst case approval vote success probability is at least $P_{\min}$ whenever the number of alternatives is equal or lower than*

$$max(\frac{(1-p_{\min})}{(2e^{-\frac{1}{2}n\Delta p^2})}+1, \frac{(1-p_{\min})(1+(n-1)\Delta p^2)}{2(1-\Delta p^2)}+1). \quad (1)$$

Algorithm 1. The simulations were conducted as depicted by Algorithm 1. Upon receiving a predetermined set of input parameters, the algorithm initiates by computing the number of bins m using theorem 1 and t. These bins, indicative of the precision achieved by VfB, are crucial for constructing the agents' beliefs. Subsequently, a random value p^* is generated, representing the true probability for the event under consideration. This probability inevitably falls within a single bin, termed the *ground truth bin*. Subsequently, the algorithm enters a for loop iterating over the number of simulation rounds. For every agent, the algorithm generates imprecise beliefs (Subroutine 2) based on precise probabilities generated to reflect the likelihood of their beliefs encompassing values from one or more bins (Subroutine 1). Further explanations of both subroutines are provided below. Upon belief construction, Algorithm 1 computes the aggregate of every considered pooling function along with its corresponding epistemic value. Ultimately, Algorithm 1 calculates the average epistemic value of each pooling function across all simulation rounds, alongside an average imprecise belief and aggregate. The latter two are computed by summing all lower and upper probabilities, respectively, and dividing by the total number of these probabilities across all agents and simulation rounds.

Subroutine 1, Algorithm 2. The first subroutine is responsible for generating precise probabilities to ensure that agents include specific bins in their imprecise beliefs, meeting the Δp condition. Recall that Δp represents the margin by which agents are, on average, more likely to vote for the ground truth bin than for any other. By construction of our algorithm and the definition of VfB, this is

Algorithm 1: Simulations with n agents.

Procedure `simulations`$(n, \Delta p, P_{\min}, t, \alpha, l)$
    $m \leftarrow \lfloor \frac{max \times t}{100} \rfloor$;
    $p^\star \leftarrow$ value from uniform distribution over $[0, 1]$ choose ground truth bin ω_* s.t. $p^\star \in \omega_*$
    for $1 \dots \ell$ **do**
        Subroutine 1: Construct precise probabilities $p_i^{\omega_j} \forall i \in \mathcal{A}, \forall j \in \mathcal{W}$;
        Subroutine 2: Construct imprecise beliefs $\mathcal{P}_i \; \forall i \in \mathcal{A}$;
        for *VfB, Linear, Log, Convex* **do**
            Compute Aggregate $\mathcal{F}(\mathcal{P}_1, ..., \mathcal{P}_n)$
            Compute Epistemic Value $\mathcal{V}(\mathcal{F}(\mathcal{P}_1, ..., \mathcal{P}_n))$
        end
    end
    for *VfB, Linear, Log, Convex* **do**
        Compute average $\mathcal{V}$: $\bar{\mathcal{V}} \leftarrow \frac{1}{l} \sum_{k=1}^{l} \mathcal{V}_k$
        Compute average $\mathcal{P}$: Let $lp1$ and $up1$ be all lower and upper probabilities for each l and each $\mathcal{P}_i$: $\bar{\mathcal{P}} \leftarrow [\frac{1}{(l \times n)} \sum_{k=1}^{(l \times n)} lp1_k, \frac{1}{(l \times n)} \sum_{k=1}^{(l \times n)} up1_k]$
        Compute average $\mathcal{F}(\mathcal{P}_1, ..., \mathcal{P}_n)$: Let $lp2$ and $up2$ be all lower and upper probabilities for each l and each $\mathcal{F}(\mathcal{P}_1, ..., \mathcal{P}_n)$: $\bar{\mathcal{F}}(\mathcal{P}_1, ..., \mathcal{P}_n) \leftarrow [\frac{1}{(l \times n)} \sum_{k=1}^{(l \times n)} lp2_k, \frac{1}{(l \times n)} \sum_{k=1}^{(l \times n)} up2_k]$
    end

equivalent to saying that an agent's belief is centered around the ground truth bin with a probability meeting the Δp condition. Beginning with the ground truth bin containing the correct probability, we initialize a target value, denoted by $\mathbb{E}(\bar{p}^{\omega_*})$, sampled from a uniform distribution ranging between Δp and 1. Here, $(\bar{p}^{\omega_*})$ signifies the average probability for the ground truth bin to be the center of the agent's belief. Subsequently, we compute the standard deviation $\sigma(\mathbb{E}(\bar{p}^{\omega_*}))$, setting it as $\frac{min(\mathbb{E}(\bar{p}), 1-\mathbb{E}(\bar{p}))}{3}$ to ensure that the precise values constructed for this target probability are centered closely around $\mathbb{E}(\bar{p}^{\omega_*})$ with a high likelihood. Given $\sigma(\mathbb{E}(\bar{p}^{\omega_*}))$, we proceed to generate precise probabilities for each agent, where each $p_i^{\omega_*}$ reflects the probability for agent i to include p^* in their belief, following a normal distribution centered around $\mathbb{E}(\bar{p}^{\omega_*})$.

In the process of constructing these values according to a normal distribution, it's possible that some outliers may fall below 0 or exceed 1. To address this, we implement a *clipping* mechanism where outliers less than 0 are set to 0 and those greater than 1 are set to 1. For VfB applicability, we must ensure that $\bar{p}^{\omega_*}$ is at least Δp. When a significant number of outliers exceeding 1 are clipped, the average of the generated precise probabilities may fall below $\mathbb{E}(\bar{p}^{\omega_*})$, potentially violating the Δp condition. Therefore, Algorithm 2 calculates the actual average value $(\bar{p}^{\omega_*})$ from the precise probabilities and verifies if the Δp condition holds. If not, a new set of precise probabilities must be generated until the condition is satisfied. If the condition is met, the procedure is repeated for all bins except the ground truth bin and all agents. In this scenario, $(\mathbb{E}(\bar{p}^{\omega_\dagger}))$ must fall between 0 and $\bar{p}^{\omega_*} - \Delta p$ to satisfy the Δp condition. Consequently, $(\mathbb{E}(\bar{p}^{\omega_\dagger}))$ is set within these bounds, and a final check ensures that the average of the actual precise values after clipping, $\bar{p}^{\omega_\dagger}$, is less than $\bar{p}^{\omega_*} - \Delta p$. If the condition is not met, all precise probabilities default to $(\mathbb{E}(\bar{p}^{\omega_\dagger}))$ to ensure expedited termination, as $(\mathbb{E}(\bar{p}^{\omega_\dagger}))$ inherently meets the Δp condition.

Algorithm 2: Construction of Precise Probabilities for n agents.

1 **Procedure** `simulations`$(n, \Delta p, P_{\min}, t, \alpha, l)$
2 $\quad \mathbb{E}(\bar{p}^{\omega_*}) \leftarrow$ value from uniform distribution over $[\Delta p, 1]$
3 $\quad \sigma(\mathbb{E}(\bar{p}^{\omega_*})) \leftarrow \frac{min(\mathbb{E}(\bar{p}^{\omega_*}),1-\mathbb{E}(\bar{p}^{\omega_*}))}{3}$
4 $\quad$ generate $p_1^{\omega_*}, ..., p_n^{\omega_*}$ following normal distribution with $\sigma(\mathbb{E}(\bar{p}^{\omega_*}))$
5 $\quad$ s.t $\forall p_i^{\omega_*} < 0 \rightarrow 0$ and $\forall p_i^{\omega_*} > 1 \rightarrow 1$;
6 $\quad \bar{p}^{\omega_*} \leftarrow \frac{1}{n} \sum_1^n p_i^{\omega_*}$;
7 $\quad$ **If** $\bar{p}^{\omega_*} \geq \Delta p$ **proceed**, **otherwise** redo lines 2-6
8 $\quad$ **for** $\forall \omega_\dagger \in \mathcal{W} \setminus \omega_*$ **do**
9 $\qquad \mathbb{E}(\bar{p}^{\omega_\dagger}) \leftarrow$ value from uniform distribution over $[0, \bar{p}^{\omega_*} - \Delta p]$
10 $\qquad$ generate $p_1^{\omega_\dagger}, ..., p_n^{\omega_\dagger}$ following normal distribution with $\sigma(\mathbb{E}(\bar{p}^{\omega_\dagger}))$ s.t $\forall p_i^{\omega_\dagger} < 0 \rightarrow 0$, $\forall p_i^{\omega_\dagger} > 1 \rightarrow 1$ and $\sigma(\mathbb{E}(\bar{p}^{\omega_\dagger})) \leftarrow \frac{min(\mathbb{E}(\bar{p}^{\omega_\dagger}),1-\mathbb{E}(\bar{p}^{\omega_\dagger}))}{3}$;
11 $\qquad$ Compute $\bar{p}^{\omega_\dagger} \leftarrow \frac{1}{n} \sum_1^n p_i^{\omega_\dagger}$;
12 $\qquad$ **If** $\bar{p}^{\omega_\dagger} \leq (\bar{p}^{\omega_*} - \Delta p)$ **proceed**,
13 $\qquad$ **otherwise** $\forall p_i^{\omega_\dagger} = \mathbb{E}(\bar{p}^{\omega_\dagger})$
14 $\quad$ **end**

Subroutine 2, Algorithm 3. In this second subroutine, we generate imprecise beliefs based on the outcomes of the first subroutine. Initially, we ensure the Δp condition is met. For each agent, a random number a is generated within the range of 0 to 1. We validate if a lies within the interval of 0 to the agent's precise probability for the ground truth bin. If affirmative, the imprecise belief is centered around the ground truth bin. Otherwise, if a falls outside this interval, the imprecise degree must be constructed from a bin other than the ground truth bin. To determine the belief center, we establish intervals for each bin, wider for bins with higher precise probabilities for the respective agent. These intervals form an ordered chain $s_1, ..., s_m$, where each interval's upper probability is the lower probability of the next, and the new upper probability is the sum of the previous upper probability and the precise value for the current bin, except that the precise value for the ground truth bin is substituted by 0. Subsequently, a random number b is generated between 0 and the sum of all precise beliefs of the agent for all bins except the ground truth bin. The interval containing b is designated as the center of the agent's imprecise belief.

In the second part of Algorithm 3, we designate the bin determined as the belief center and denote it as ω_j. The algorithm then iterates over all bins ω_l

Algorithm 3: Construction of Imprecise Beliefs.

Procedure `simulations`$(n, \Delta p, P_{\min}, t, \alpha, l)$
    for $1 \ldots n$ **do**
        Generate random number a following uniform distribution over $[0, 1]$
        If $a \in [0, p_i^{\omega_*}]$, **then** construct belief center of $\mathcal{P}_i$ from ω_*, **otherwise**
        $\forall \omega_\dagger \in \mathcal{W}$ form intervals of the form
        $[0, p_i^{\omega_1}], [p_i^{\omega_1}, p_i^{\omega_1} + p_i^{\omega_2}], ..., [\sum_{\omega_1}^{\omega_{m-1}}, \sum_{\omega_1}^{\omega_m}] = s_1, ..., s_j, ..., s_m$ with $p_i^{\omega_*} \overset{!}{=} 0$.
        generate random number b from $[0, \sum_{\omega_1}^{\omega_{m-1}}]$
        if $b \in s_j$ **then**
            Construct belief center of $\mathcal{P}_i$ from ω_j
        end
        Fix ω_j as belief center.
        for $\forall \omega_l \in \mathcal{W} \setminus \omega_j$ **do**
            Generate random number from c from $[0, 1]$.
            if $c \in [0, p_i^{\omega_l}]$ **then**
                Save (a_i, ω_l)
            end
            if *there exists a path from* ω_j *to* ω_l *such that* $\forall \omega_t$ *in between* j, l, *we have* (a_i, ω_t) **then**
                $\mathcal{P} = [\omega_{j_l}, \omega_{j_r}]$ **otherwise** $\mathcal{P} = \omega_j = [a_j, a_{j+1}]$.
            end
        end
    end

other than ω_j and generates a random number c from the unit interval. For each bin, it checks if c falls into the interval $[0, p_i^{\omega_l}]$. If affirmative, it records (a_i, ω_l), indicating that if the imprecise degree of belief were not convex, values from this bin would be included in the agent's imprecise belief. Considering the convexity requirement, the algorithm proceeds to verify if a path of bins exists from ω_j to its left and right, where each bin ω_t along this path satisfies (a_i, ω_t). Should such a path exist, the agent's imprecise belief is the interval $[\omega_{jl}, \omega_{jr}]$, where ω_{jl} is the lower probability of the leftmost included bin and ω_{jr} is the upper probability of the rightmost included bin. If no such path exists, the imprecise belief defaults to the whole width of the center bin.