

ERPM — Avancement Wrapper Et Nouveaux Effets

Jérémie Chichignoud (CUB'iTECH)

18 novembre 2025

Table des matières

Objectifs (portée du document)	3
Travail réalisé	3
Travail restant et perspectives	4
Tableau de correspondance des effets	5
Explication effet par effet	7
1 Comptage de groupes par taille : <code>groups(from, to)</code>	7
2 Somme des puissances de tailles : <code>squared_sizes(from, to, pow)</code>	7
3 k-cliques sur la projection 1-mode : <code>cliques(k, normalized)</code>	8
4 Cliques géométriquement pondérées : <code>cliques_GW(λ)</code>	8
5 Somme des logs de factorielles décalées : <code>log_factorial_sizes</code>	10
6 Effet <code>cov_ingroup</code>	12
7 Effet <code>cov_fullmatch</code>	14
8 Homophilie discrète par cliques : <code>cov_match(k)</code>	15
9 Homophilie géométriquement pondérée : <code>cov_match_GW(λ)</code>	16
10 Effet <code>cov_fulldiff</code>	17
11 Dispersion intra-groupe sur attribut numérique : <code>cov_diff</code>	18

12	Dispersion multi-échelle sur attribut numérique : <code>cov_diff_GW</code>	20
13	Covariée dyadique intra-groupe : <code>dyadcov_full</code>	21
14	Covariée dyadique sur cliques intra-groupe : <code>dyadcov</code>	23
Annexe — ERGM, Metropolis–Hastings, vraisemblance et critères d’ajustement		26
A.1	Rappel : qu’est-ce qu’un ERGM ?	26
A.2	Log-vraisemblance, déviance, AIC et BIC	27
A.3	Sorties usuelles de <code>summary(ergm)</code>	28
A.4	Contraintes et mécanisme de proposition (MH) en biparti	28
A.5	Exemple minimal (R)	28
A.6	Schéma du processus d’estimation	29
A.7	Points d’attention	29

État d'avancement : wrapper `erpm` et statistiques/effets

Objectifs (portée du document)

L'objectif est de concevoir une fonction `erpm()` inspirée de la logique des modèles linéaires généralisés, capable d'estimer des effets statistiques définis sur des **partitions** plutôt que sur des graphes. La syntaxe cible est :

```
erpm(partition ~ effets + options)
```

La partie gauche représente la partition des acteurs en groupes, et la partie droite décrit les effets statistiques à calculer.

La fonction agit comme un **wrapper** autour du moteur `ergm()`. Elle transforme la partition en un **graphe biparti** (acteurs-groupes), sur lequel `ergm()` peut être appliqué directement. Cette traduction permet d'utiliser les outils et algorithmes d'ERGM tout en conservant une lecture centrée sur les groupes.

Un autre rôle clé de `erpm()` est d'assurer la **correspondance entre effets ERPM et ERGM**, selon trois situations :

1. **Effet direct** : déjà présent dans ERGM et utilisable tel quel ;
2. **Effet partiel** : partiellement couvert, nécessitant un traitement complémentaire (agrégation, normalisation, etc.) ;
3. **Effet spécifique** : absent d'ERGM, à créer avec `InitErgmTerm()` et une fonction C de changement (`change_stat`).

L'objectif final est que `erpm()` pilote l'ensemble du processus : de la définition de la partition à l'appel de `ergm()`, en passant par la préparation des effets, des contraintes et des propositions.

Travail réalisé

La structure principale de `erpm()` est **fonctionnelle et stable**. Elle gère correctement la conversion des entrées, le renommage des effets et l'appel au moteur `ergm()`.

- **Structure du wrapper** : prise en charge d'une partition et d'une formule d'effets ; construction de la matrice d'affiliation (acteurs \times groupes) ; création du **network** biparti ; ajout automatique des contraintes et propositions nécessaires (`InitErgmConstraint.bipart`, `InitErgmProposal.BiPart`) ; puis exécution de `ergm()`.
- **Traduction des effets** : les effets connus sont traduits vers leurs équivalents ERGM (par ex. `groups` \rightarrow `b2degrange`). Les effets dyadiques utilisent les opérateurs `Proj1` ou `B` selon le mode d'analyse.
- **Effets implémentés** : un effet spécifique, `squared_sizes`, est pleinement fonctionnel. Il dispose de sa fonction `InitErgmTerm()` et de son code C associé, basé sur les macros d'ERGM (`ergm_changestat_common.do_not_include_directly.h`, `ergm_changestat.h`, etc.).
- **Tests et validation** : les scripts `squared_sizes.R` et `test_groups.R` vérifient la cohérence du pipeline complet : création de partitions simples, conversion bipartie, appel à `erpm()`, et calcul des statistiques. Ces tests sont conçus comme des *selftests* autonomes.

Les résultats montrent que l'intégration avec `ergm()` fonctionne correctement et que la structure du wrapper est fiable.

Travail restant et perspectives

La prochaine étape est la **caractérisation complète des effets ERPM**. Chaque effet doit être identifié comme direct, partiel ou spécifique, et implémenté en conséquence, avec un test unitaire dédié.

- **Finalisation du wrapper** : améliorer la gestion des options `control.ergm`, définir des valeurs par défaut cohérentes, traiter les cas limites (groupes vides, partitions incomplètes), et renforcer la traduction automatique des effets directs `erpm` \rightarrow `ergm`.
- **Validation des effets et tests d'intégration** : ajouter un *selftest* autonome pour chaque nouvel effet, et construire des cas minimaux pour vérifier

la stabilité statistique (moyennes, variances, convergence).

- **Effets liés aux attributs dyadiques** : une partition ne comporte pas d'attributs dyadiques, mais il peut être utile de modéliser la probabilité d'appartenance à un groupe à partir d'informations dyadiques externes. Dans ce cas, une matrice d'attributs dyadiques peut être transmise en complément de la partition.
- **Documentation** : compléter les blocs `roxygen2` et rédiger une vignette “`erpm` en dix lignes” illustrant le flux complet : partition \rightarrow graphe biparti \rightarrow estimation \rightarrow résumé.

En résumé, `erpm()` est stable, bien intégré à `ergm()`, et prêt à accueillir progressivement de nouveaux effets ERPM.

Tableau de correspondance des effets

Effet ERPM	Correspondance ERGM	Alias/Terms ERGM	Calcul partiel (si partielle)	Si aucune : InitErgmTerm + C	Commentaire	Status
<code>groups(from,to)</code>	Directe	<code>b2degrange(from,to)</code>	—	—	Taille des groupes = degré (mode 2).	✓
<code>squared_sizes(from,to, pow)</code>	ErgmTerm	—	—	<code>InitErgmTerm.squared_sizes</code> <code>c_squared_sizes</code>	Somme des tailles des groupes (mode 2) élevées à la puissance <code>pow</code> .	✓
<code>log_factorial_sizes</code>	ErgmTerm	—	—	<code>InitErgmTerm.log_factorial_sizes</code> <code>c_log_factorial_sizes</code>	À remplir	✓
<code>cliques(clique_size, normalized)</code>	ErgmTerm	—	—	<code>InitErgmTerm.cliques</code> <code>c_cliques</code>	Compte, pour chaque groupe du mode 2, le nombre de k -cliques d'acteurs qu'il induit dans la projection 1-mode ; chaque groupe de taille n_g contribue $\binom{n_g}{k}$.	✓
<code>cliques_GW(lambda)</code>	ErgmTerm	—	—	<code>InitErgmTerm.cliques_GW</code> <code>c_cliques_GW</code>	À remplir	✓
<code>cov_ingroup(cov,size, category)</code>	ErgmTerm	—	—	<code>InitErgmTerm.cov_ingroup</code> <code>c_cov_ingroup</code>	À remplir	✓
<code>cov_match(cov)</code>	ErgmTerm	—	—	<code>InitErgmTerm.cov_match</code> <code>c_cov_match</code>	À remplir	✓
<code>cov_match_GW(cov, lambda)</code>	ErgmTerm	—	—	<code>InitErgmTerm.cov_match_GW</code> <code>c_cov_match_GW</code>	À remplir	✓
<code>cov_fullmatch(cov)</code>	ErgmTerm	—	—	<code>InitErgmTerm.cov_fullmatch</code> <code>c_cov_fullmatch</code>	À remplir	✓

Effet ERPM	Correspondance ERGM	Alias/Termes ERGM	Calcul partiel (si partielle)	Si aucune : InitErgmTerm + C	Commentaire	Status
<code>cov_diff(cov)</code>	ErgmTerm	—	—	<code>InitErgmTerm.cov_diff</code> <code>c_cov_diff</code>	À remplir	✓
<code>cov_diff_GW(cov)</code>	ErgmTerm	—	—	<code>InitErgmTerm.cov_diff_GW</code> <code>c_cov_diff_GW</code>	À remplir	✓
<code>cov_fulldiff(cov)</code>	ErgmTerm	—	—	<code>InitErgmTerm.cov_fulldiff</code> <code>c_cov_fulldiff</code>	À remplir	✓
<code>dyadcov(dyadcov, clique_size, normalized)</code>	ErgmTerm	—	—	—	À remplir	✗
<code>dyadcov_full(dyadcov, size)</code>	ErgmTerm	—	—	—	À remplir	✓
<code>dyadcov_GW(dyadcov, lambda)</code>	ErgmTerm	—	—	—	À remplir	✗

Explication effet par effet

1 Comptage de groupes par taille : `groups(from, to)`

Définition.

Pour le biparti $A-G$ et $n_g = d(g)$ la taille du groupe g ,

$$\text{groups}(\text{from}, \text{to}) = \sum_{g \in G} \mathbf{1}\{n_g \in [\text{from}, \text{to})\},$$

Cela compte le nombre de groupes dont la taille appartient à l'intervalle $[\text{from}, \text{to})$ (borne inférieure incluse, supérieure exclue).

Correspondance ERGM.

`groups(from,to) → b2degrange(from=from, to=to)` sur le *mode 2*.

Appel dans `erpm()`.

```
1 # Tous les groupes de taille ≥1
2 erpm(partition ~ groups)
3
4 # Groupes de taille exactement 3 -> [3,4)
5 erpm(partition ~ groups(3))
6
7 # Intervalle explicite [2,5)
8 erpm(partition ~ groups(from = 2, to = 5))
```

2 Somme des puissances de tailles : `squared_sizes(from, to, pow)`

Définition.

Sur les nœuds groupes $g \in G$ de taille n_g ,

$$\text{squared_sizes}(\text{from}, \text{to}, \text{pow}) = \sum_{g \in G} \mathbf{1}\{n_g \in [\text{from}, \text{to})\} n_g^{\text{pow}},$$

Par défaut $\text{pow} = 2$. Le terme est vectorisable sur plusieurs intervalles et puissances.

Implémentation ERGM dédiée.

Terme spécifique `squared_sizes` côté `{ergm}`, calcul en « un-toggle » sur le mode 2 uniquement, avec entrées $[\text{from}, \text{to}, \text{pow}]$ et gestion de $[\text{from}, \text{to})$ ($\text{to} = \infty \Rightarrow$ remplacé par $n + 1$).

Appel dans `erpm()`.

```
1 # Somme des carrés pour tous les groupes (from=1, to=Inf, pow
  =2)
2 erpm(partition ~ squared_sizes)
3
4 # Somme des cubes pour tailles [2,5)
5 erpm(partition ~ squared_sizes(from = 2, to = 5, pow = 3))
6
7 # Vectorisation sur deux intervalles
8 erpm(partition ~ squared_sizes(from = c(1,4), to = c(4, Inf),
  pow = c(2,2)))
```

3 k-cliques sur la projection 1-mode : cliques(k, normalized)

Cliques d'acteurs via la projection 1-mode.

Pour une partition P , on construit le graphe biparti $A-G$ où A est l'ensemble des acteurs et G l'ensemble des groupes. Pour tout $g \in G$, on note $n_g = d(g)$ le degré (i.e. la taille) du groupe g . On note $N = \{n_g : g \in G\}$ l'ensemble des tailles de groupes.

La projection 1-mode sur A relie deux acteurs s'ils sont co-membres d'au moins un groupe. Le nombre total de k -cliques d'acteurs dans cette projection vaut :

$$\#k\text{-cliques} = \sum_{g \in G} \binom{n_g}{k},$$

où l'on adopte la **convention combinatoire étendue par zéro** :

$$\binom{n}{k} = \begin{cases} \frac{n!}{k!(n-k)!}, & \text{si } n \geq k \geq 0, \\ 0, & \text{sinon.} \end{cases}$$

Ainsi, les groupes de taille $n_g < k$ ne contribuent pas au comptage. S'il n'existe aucun groupe de taille au moins k , alors $\#k\text{-cliques} = 0$.

Exemple. Pour la partition $P = \{1, 1, 1, 1, 2, 2, 3\}$, les tailles de groupes sont $N = \{4, 2, 1\}$. On obtient :

$$\#2\text{-cliques} = \binom{4}{2} + \binom{2}{2} + \binom{1}{2} = 6 + 1 + 0 = 7,$$

$$\#4\text{-cliques} = \binom{4}{4} + \binom{2}{4} + \binom{1}{4} = 1 + 0 + 0 = 1,$$

Ainsi, l'ensemble des tailles de cliques présentes est $K = \{2, 4\}$.

Appel dans `erpm()`.

```
1 # k par défaut = 2, non normalisé
2 erpm(partition ~ cliques())
3
4 # k explicite
5 erpm(partition ~ cliques(k = 3))
6
7 # Forme abrégée
8 erpm(partition ~ cliques(4))
9
10 # Normalisation par C(N1, k)
11 erpm(partition ~ cliques(k = 2, normalized = TRUE))
```

4 Cliques géométriquement pondérées : cliques_GW(λ)

Définition combinatoire.

Pour une partition P et le graphe biparti $A-G$ associé, chaque groupe $g \in G$ de taille $n_g = d(g)$ génère, dans la projection 1-mode sur A , un ensemble de cliques d'acteurs :

$$c_k(P) = \sum_{g \in G} \binom{n_g}{k}, \quad k \geq 1,$$

Ces $c_k(P)$ comptent le nombre total de k -cliques d'acteurs issus de la coappartenance à un même groupe (voir 3).

L'effet `cliques_GW(lambda)` combine ces contributions pour toutes les tailles $k \geq 1$

à l'aide d'une **série géométrique à décroissance contrôlée par λ** :

$$\begin{aligned} \text{cliques_GW}(\lambda) &= \sum_{k \geq 1} \left(-\frac{1}{\lambda}\right)^{k-1} c_k(P) \\ \text{avec } c_k(P) &= \sum_{g \in G} \binom{n_g}{k} \Rightarrow \sum_{k \geq 1} \left(-\frac{1}{\lambda}\right)^{k-1} \sum_{g \in G} \binom{n_g}{k} \\ (\text{sommes finies : } \#G < \infty, n_g < \infty) &\Rightarrow \sum_{g \in G} \sum_{k \geq 1} \left(-\frac{1}{\lambda}\right)^{k-1} \binom{n_g}{k} \\ (\text{convention } \binom{n}{k} &= 0 \text{ pour } k > n) \Rightarrow \sum_{g \in G} \sum_{k=1}^{n_g} \left(-\frac{1}{\lambda}\right)^{k-1} \binom{n_g}{k}, \end{aligned}$$

On introduit alors la contribution individuelle d'un groupe g :

$$S(n_g, \lambda) = \sum_{k=1}^{n_g} \left(-\frac{1}{\lambda}\right)^{k-1} \binom{n_g}{k},$$

La statistique globale s'écrit alors simplement :

$$\boxed{\text{cliques_GW}(\lambda) = \sum_{g \in G} S(n_g, \lambda),}$$

Expression de $S(n_g, \lambda)$ sous forme fermée.

On part de la définition combinatoire

$$S(n_g, \lambda) = \sum_{k=1}^{n_g} \left(-\frac{1}{\lambda}\right)^{k-1} \binom{n_g}{k},$$

et on cherche une expression explicite sous *forme fermée*. En posant $r = -1/\lambda$, on obtient :

$$S(n_g, \lambda) = \sum_{k=1}^{n_g} r^{k-1} \binom{n_g}{k},$$

On multiplie par r pour aligner les puissances :

$$r S(n_g, \lambda) = \sum_{k=1}^{n_g} r^k \binom{n_g}{k},$$

Puis on complète la somme par le terme $k = 0$ que l'on retranche aussitôt :

$$r S(n_g, \lambda) = \sum_{k=0}^{n_g} \binom{n_g}{k} r^k - \binom{n_g}{0} r^0 = \sum_{k=0}^{n_g} \binom{n_g}{k} r^k - 1,$$

On applique ensuite la **formule du binôme de Newton**, $(x + y)^n = \sum_{k=0}^n \binom{n}{k} x^k y^{n-k}$, avec $x = r$ et $y = 1$, d'où $(1 + r)^{n_g} = \sum_{k=0}^{n_g} \binom{n_g}{k} r^k$. On en déduit :

$$r S(n_g, \lambda) = (1 + r)^{n_g} - 1 \Rightarrow S(n_g, \lambda) = \frac{(1 + r)^{n_g} - 1}{r},$$

Enfin, la substitution $r = -1/\lambda$ conduit à la forme fermée recherchée :

$$\boxed{S(n_g, \lambda) = \lambda \left[1 - \left(\frac{\lambda-1}{\lambda} \right)^{n_g} \right]},$$

Par sommation sur tous les groupes $g \in G$, on obtient la statistique globale :

$$\text{cliques_GW}(\lambda) = \sum_{g \in G} S(n_g, \lambda) = \sum_{g \in G} \lambda \left[1 - \left(\frac{\lambda-1}{\lambda} \right)^{n_g} \right],$$

Interprétation.

Chaque groupe g de taille n_g contribue une valeur $S(n_g, \lambda)$ qui agrège l'ensemble de ses cliques potentielles, pondérées par la série géométrique $(-1/\lambda)^{k-1}$. Le terme est donc une *mesure continue de densité de coappartenance*, où λ agit comme un facteur de décroissance. L'effet **cliques_GW**(λ) quantifie, de manière lissée et graduelle, à quel point les acteurs appartiennent ensemble à des groupes communs.

Cas particuliers et limites.

- **Cas $\lambda = 1$** : Les poids deviennent $(-1)^{k-1}$. On obtient $S(n_g, 1) = \sum_{k=1}^{n_g} (-1)^{k-1} \binom{n_g}{k} = 1$. Chaque groupe non vide contribue exactement 1. L'effet `cliques_GW(1)` compte donc simplement le nombre total de groupes, indépendamment de leur taille.
- **Cas $\lambda \rightarrow \infty$** : $(1 - 1/\lambda)^{n_g} \approx 1 - n_g/\lambda$, d'où $S(n_g, \lambda) \approx n_g/\lambda$. Les grandes cliques deviennent négligeables : seule la composante linéaire ($k = 1$) subsiste. L'effet tend alors vers une mesure proportionnelle au nombre total d'acteurs.
- **Cas λ petit (proche de 1)** : Les poids $(-1/\lambda)^{k-1}$ décroissent lentement, les cliques de grande taille sont fortement prises en compte. Le terme devient sensible à la concentration d'acteurs dans des groupes volumineux.

Résumé interprétatif.

Régime de λ	Formule dominante	Interprétation intuitive
$\lambda = 1$	$S(n_g, 1) = 1$	Chaque groupe vaut 1 (compte des groupes)
$\lambda \rightarrow \infty$	$S(n_g, \lambda) \approx n_g/\lambda$	Poids linéaire des acteurs (1-cliques)
λ petit	Série complète sur k	Forte sensibilité aux cliques denses

Appel dans `erpm()`.

```

1 # Forme par défaut : lambda = 2
2 erpm(partition ~ cliques_GW)
3
4 # Lambda explicite
5 erpm(partition ~ cliques_GW(lambda = 3))
6
7 # Vectorisation (plusieurs décroissances)
8 erpm(partition ~ cliques_GW(lambda = c(1.5, 2, 4)))

```

1. Inexistant pour les partitions car les groupes y sont disjoints et indépendants.

Remarque.

L'effet `cliques_GW` est à la fois un *lissage analytique* du comptage discret des `cliques(k)` et une *pondération géométrique* contrôlant la profondeur des interactions denses. Il fournit un indicateur continu de la cohésion interne des groupes à travers leurs cliques d'acteurs.

5 Somme des logs de factorielles décalées : `log_factorial_sizes`

Définition combinatoire.

Pour une partition P de N acteurs en groupes $g \in G$, on note $n_g = |g|$ la taille (ou degré mode-2) de chaque groupe. L'effet `log_factorial_sizes` associe à chaque groupe le logarithme du *factoriel décalé* de sa taille :

$$S(P) = \sum_{g \in G} \log((n_g - 1)!) = \sum_{g \in G} \text{lgamma}(n_g),$$

La fonction $\text{lgamma}(n) = \log(\Gamma(n)) = \log((n-1)!)$ est la version continue du logarithme factoriel, numériquement stable et définie pour tout $n > 0$. Les groupes vides ($n_g = 0$) sont exclus ou traités par convention $f(0) = 0$.

Cette forme additive sur les groupes exprime une **mesure combinatoire interne** à chaque groupe, c'est-à-dire une quantité fondée sur le nombre de façons différentes d'organiser les membres *à l'intérieur* du groupe, sans tenir compte des relations entre groupes¹. Autrement dit, on ne considère pas combien il existe de groupes ni comment ils interagissent, mais uniquement combien de configurations distinctes un seul groupe peut avoir selon sa taille n_g .

Ici, la mesure associée à un groupe est $(n_g - 1)!$, le nombre d'**arrangements circulaires** possibles de ses n_g membres :

$$(n_g - 1)! \text{ ordres distincts sur un cercle.}$$

Un *arrangement circulaire* (ou *permutation circulaire*) désigne la manière de placer n objets différents autour d'un cercle, où les rotations sont considérées comme équivalentes :

- Sur une ligne, on compte $n!$ permutations distinctes.
- Sur un cercle, les n rotations d'une même disposition sont indiscernables, donc le nombre d'arrangements distincts est $n!/n = (n-1)!$.

Ainsi, $(n_g - 1)!$ représente le nombre de façons uniques de disposer les membres du groupe sur un cercle, modulo les rotations. Le logarithme de cette quantité, $\log((n_g - 1)!)$, mesure la complexité combinatoire interne du groupe.

La statistique globale de la partition s'interprète alors comme le *log-produit* de ces complexités internes :

$$S(P) = \log \left(\prod_{g \in G} (n_g - 1)! \right),$$

Elle résume la richesse combinatoire totale des organisations internes de tous les groupes.

Expression bipartie (ERPM \rightarrow ERGM).

Dans la traduction ERPM, on associe au partitionnement un graphe biparti (A, G, E) où :

- A désigne l'ensemble des **acteurs** (ou éléments de la partition) ;
- G l'ensemble des **groupes** de la partition ;
- $E \subseteq A \times G$ l'ensemble des **liens d'appartenance**, chaque arête $(i, g) \in E$ indiquant que l'acteur i appartient au groupe g .

Le degré $d(g)$ d'un nœud $g \in G$ (nombre d'arêtes incidentes sur g) correspond exactement à la taille du groupe : $d(g) = n_g$.

La statistique se réécrit donc :

$$\text{log_factorial_sizes} = \sum_{g \in G} \lgamma(d(g)),$$

soit une *somme de fonctions du degré des nœuds du mode 2*. Chaque nœud-groupe contribue proportionnellement à la complexité combinatoire de ses liens d'appartenance.

Développement asymptotique.

En appliquant l'approximation de Stirling, $\log((n-1)!) \approx (n - \frac{1}{2}) \log n - n + \frac{1}{2} \log(2\pi)$, on obtient :

$$S(P) \approx \sum_{g \in G} \left[(n_g - \frac{1}{2}) \log n_g - n_g \right] + C,$$

Cette expression relie l'effet à la *distribution des tailles de groupes*. En posant $p_g = n_g/N$,

$$S(P) \approx N \sum_g p_g \log p_g + N \log N + C',$$

À somme N fixée, S augmente lorsque la distribution (p_g) devient plus inégale : il favorise les regroupements concentrés (faible entropie).

Interprétation.

- **Combinatoire interne** : chaque groupe contribue au log du nombre de permutations circulaires possibles de ses membres. La statistique globale quantifie la “complexité interne” moyenne des groupes de la partition.
- **Lien entropique** : en régime large (n_g grand), la dérivée locale $\partial S / \partial n_g \approx \log n_g$ augmente doucement avec la taille, d'où un effet de *préférence logarithmique* pour les grands groupes. Une estimation positive de son paramètre θ favorise donc des regroupements plus concentrés.
- **Comparaison avec d'autres effets** :

Effet	Formule	Croissance
squared_sizes	$\sum n_g^2$	quadratique ($\propto n^2$)
log_factorial_sizes	$\sum \lgamma(n_g)$	$n \log n$ (intermédiaire)
groups	$\sum 1_{n_g > 0}$	constante

Cas asymptotiques.

- **Tous les groupes singletons** : $n_g = 1 \Rightarrow S = 0$.
- **Un seul grand groupe** : $S = \log((N-1)!)$ maximal.
- **Partition équilibrée** : pour un nombre total de groupes K fixé et un nombre d'acteurs N constant, la valeur de $S = \sum_g \text{lgamma}(n_g)$ est **minimale lorsque les tailles des groupes sont aussi égales que possible**, c'est-à-dire $n_g \simeq N/K \forall g$.

En effet, la fonction $\text{lgamma}(n)$ est convexe en n , donc par l'inégalité de Jensen :

$$\frac{1}{K} \sum_g \text{lgamma}(n_g) \geq \text{lgamma}\left(\frac{1}{K} \sum_g n_g\right) = \text{lgamma}(N/K),$$

L'égalité (et donc le minimum de S) est atteinte uniquement lorsque tous les groupes ont la même taille.

Intuitivement, plus les tailles des groupes sont déséquilibrées, plus certaines valeurs n_g deviennent grandes, et comme $\text{lgamma}(n)$ croît non-linéairement ($\sim n \log n$), la somme totale S augmente. La partition la plus homogène minimise donc la “complexité combinatoire interne” globale.

Résumé analytique.

$$\text{log_factorial_sizes}(P) = \sum_{g \in G(P)} \text{lgamma}(|g|) \quad \text{où} \quad \text{lgamma}(n) = \log((n-1)!),$$

C'est une statistique additive sur les groupes, convexe en (n_g) , croissant plus lentement que n_g^2 , et reliée à la dispersion des tailles par approximation de Stirling. Elle n'a pas d'équivalent direct dans **ergm**, mais peut être implémentée comme un terme spécifique de type “somme de fonctions du degré (mode 2)”.

Remarque finale.

L'effet `log_factorial_sizes` introduit un contrôle logarithmique du déséquilibre des tailles de groupes. Son comportement se situe entre la linéarité de `groups` et la convexité forte de `squared_sizes`.

Appel dans `erpm()`.

```
1 erpm(partition ~ log_factorial_sizes)
```

6 Effet `cov_ingroup`

Cadre.

On partitionne des acteurs $A = \{1, \dots, N\}$ en groupes G . Le biparti associé est $B = (A, G, E)$. Chaque acteur i appartient à un unique groupe $g(i) \in G$. La taille d'un groupe g est $n_g = \deg_B(g) = |g|$.

Attribut et filtre de taille.

On fixe un attribut d'acteur $x = (x_i)_{i \in A}$: numérique (p.ex. âge, score) ou binaire issu d'une modalité catégorielle ciblée ($x_i = \mathbf{1}[c_i = \kappa]$). On choisit un ensemble de tailles admissibles $S \subset \mathbb{N}_{\geq 1}$ (p.ex. $S = \{k\}$, $S = \{a, \dots, b\}$, ou $S = \mathbb{N}_{\geq 1}$).

Définition.

La statistique `cov_ingroup` pondère la somme d'attributs intra-groupe par la taille du groupe, en ne gardant que les tailles dans S :

$$\begin{aligned} T(B; x, S) &= \sum_{i \in A} x_i n_{g(i)} \mathbf{1}[n_{g(i)} \in S] \\ &= \sum_{g \in G} \left(n_g \sum_{i \in g} x_i \right) \mathbf{1}[n_g \in S], \end{aligned}$$

Forme matricielle.

Soit $M \in \{0,1\}^{N \times |G|}$ la matrice d'incidence ($M_{ig} = 1 \iff i \in g$), $\mathbf{1}_N = (1, \dots, 1)^\top$, $n = M^\top \mathbf{1}_N \in \mathbb{N}^{|G|}$, $s = M^\top x \in \mathbb{R}^{|G|}$, et $w = (\mathbf{1}[n_g \in S])_{g \in G} \in \{0,1\}^{|G|}$. Alors

$$T(B; x, S) = (w \odot n)^\top s = (w \odot n)^\top M^\top x = x^\top M \text{Diag}(w \odot n) \mathbf{1}_{|G|},$$

Change-stat pour un déplacement $i : a \rightarrow b$.

Notons $X_g = \sum_{j \in g} x_j$. Avant le déplacement : tailles n_a, n_b et sommes X_a, X_b . Après : $n'_a = n_a - 1$, $X'_a = X_a - x_i$, $n'_b = n_b + 1$, $X'_b = X_b + x_i$. La variation vaut

$$\Delta T = [n'_b X'_b \mathbf{1}[n'_b \in S] - n_b X_b \mathbf{1}[n_b \in S]] + [n'_a X'_a \mathbf{1}[n'_a \in S] - n_a X_a \mathbf{1}[n_a \in S]],$$

Sans restriction de taille ($S = \mathbb{N}_{\geq 1}$),

$$\Delta T = (X_b - X_a) + (n_b - n_a + 2) x_i,$$

Exemple court.

$n_a = 3$, $X_a = 5$; $n_b = 2$, $X_b = 4$; $x_i = 2$.

- Sans filtre : $\Delta_b = (2+1)(4+2) - 2 \cdot 4 = 10$, $\Delta_a = (3-1)(5-2) - 3 \cdot 5 = -9$, donc $\Delta T = 1$.
- Avec $S = \{3, 4, \dots\}$: avant $T = 3 \cdot 5 = 15$ (seul a compte), après $T = 3 \cdot 6 = 18$ (seul b compte), donc $\Delta T = 3$.

Lecture.

Si x est binaire, $\sum_{i \in g} x_i$ est un décompte intra-groupe de la modalité ciblée et $n_g \sum_{i \in g} x_i$ mesure «taille \times effectif visé», restreint aux tailles S .

Cas limites et comportement.

- **Groupes vides.** Non définis dans un biparti valide ; la fonction les ignore.
- **Singletons.** Si $n_g = 1$, la contribution vaut $x_i \mathbf{1}[1 \in S]$ pour l'unique acteur $i \in g$.
- **Attributs manquants.** Les valeurs NA dans x sont propagées comme zéro pour éviter des contributions incohérentes ; elles doivent idéalement être nettoyées en amont.
- **Échelle de x .** Si x est numérique à forte variance, le terme peut dominer la log-vraisemblance ; il est souvent recommandé de centrer ou normaliser x .
- **Filtrage de tailles.** Si tous les $n_g \notin S$, la statistique vaut 0 ; c'est un cas limite normal mais à signaler (aucune contrainte informative).
- **Coût algorithmique.** La mise à jour locale se fait en $\mathcal{O}(1)$: seul le groupe affecté par un toggle est recalculé.
- **Dégénérescence.** Si x est quasi constant, la statistique est quasi linéaire et peut induire une estimation peu informative ; l'effet reste stable mais peu discriminant.

Exemple `erpm()` minimal.

```
1 # Partition simple
2 partition <- c(1,1, 2,2,2, 3,3,3, 4,4)
3
4 # Attributs: numérique et catégoriel
5 nodes <- data.frame(label = paste0("A", 1:10),
6                       age = c(20,30, 40,50,60, 10,10,10, 5,5),
7                       dept = c("A","B", "A","C","A", "A","A","B",
8                                "C","A"),
9                       stringsAsFactors = FALSE)
10
11 # Cas 1 : covariée numérique
12 erpm(partition ~ cov_ingroup("age", size = 2:3),
13       nodes = nodes)
14
15 # Cas 2 : covariée catégorielle avec filtrage
```

```

15 erpm(partition ~ cov_ingroup("dept", category = "A", size = 3)
16     , nodes = nodes)
17
18 # Vérification par summary sur le biparti sous-jacent
19 dry <- erpm(partition ~ cov_ingroup("age", size = 2:3),
20     eval_call = FALSE, nodes = nodes)
21 summary(dry[[2]], constraints = ~ bipart)

```

7 Effet cov_fullmatch

Cadre.

Acteurs $A = \{1, \dots, N\}$ partitionnés en groupes G . Biparti $B = (A, G, E)$. Taille du groupe $g : n_g = |g|$.

Attribut et filtre de tailles.

Attribut catégoriel $c = (c_i)_{i \in A}$ à valeurs dans un ensemble fini L (modalités). Filtre de tailles admissibles $S \subset \mathbb{N}_{\geq 1}$ ($S = \{k\}$, $S = \{a, \dots, b\}$, ou $S = \mathbb{N}_{\geq 1}$).

Définition.

Un groupe est *homogène* si tous ses membres partagent la même modalité. La statistique compte les groupes homogènes dont la taille est dans S :

$$T(B; c, S) = \sum_{g \in G} \mathbf{1}[n_g \in S] \mathbf{1}[\exists r \in L : n_{g,r} = n_g]$$

où $n_{g,r} = |\{i \in g : c_i = r\}|$. Variante ciblée (**category**= κ) :

$$T_\kappa(B; c, S) = \sum_{g \in G} \mathbf{1}[n_g \in S] \mathbf{1}[n_{g,\kappa} = n_g],$$

Cas des singletons : si $n_g = 1$ et c_i défini, le groupe compte.

Écriture par paires (projection mode 1).

Chaque groupe g induit une clique K_{n_g} entre ses membres. On a l'équivalence

$$\sum_{r \in L} \binom{n_{g,r}}{2} = \binom{n_g}{2} \iff \max_{r \in L} n_{g,r} = n_g,$$

c'est-à-dire « toutes les paires concordent sur c » \Leftrightarrow « unanimité dans g ».

Forme matricielle.

Matrice d'incidence $M \in \{0, 1\}^{N \times |G|}$, tailles $n = M^\top \mathbf{1}_N$. Pour chaque modalité r , vecteur indicateur $x^{(r)} \in \{0, 1\}^N$ et sommes par groupe $s^{(r)} = M^\top x^{(r)}$. Masque de tailles $w_g = \mathbf{1}[n_g \in S]$. Alors

$$T(B; c, S) = \sum_{g=1}^{|G|} w_g \mathbf{1}[\max_r s_g^{(r)} = n_g], \quad T_\kappa(B; c, S) = \sum_{g=1}^{|G|} w_g \mathbf{1}[s_g^{(\kappa)} = n_g],$$

Exemple minimal.

Soit trois groupes g_1, g_2, g_3 de tailles $(2, 3, 2)$ et une covariée catégorielle c :

$$g_1 = (A, A), \quad g_2 = (A, B, B), \quad g_3 = (B, B),$$

Alors

$$T(B; c, \{2, 3\}) = 2 \quad (\text{groupes } g_1 \text{ et } g_3), \quad T_A(B; c, \{2, 3\}) = 1 \quad (\text{seul } g_1),$$

Change-stat locale.

Un toggle (i, g) n'affecte que g . Avant : $F_g = \mathbf{1}[n_g \in S] \mathbf{1}[\exists r : n_{g,r} = n_g]$. Après le toggle : F'_g avec mise à jour de n_g et du n_{g,c_i} . Variation locale :

$$\Delta T = F'_g - F_g,$$

Version ciblée : $\Delta T = F_g^{(\kappa)'} - F_g^{(\kappa)}$ avec $F_g^{(\kappa)} = \mathbf{1}[n_g \in S] \mathbf{1}[n_{g,\kappa} = n_g]$.

Effets de bord et complexité.

- **Groupes vides ou hors plage.** Les groupes de taille 0 n'existent pas dans le biparti normalisé. Si $n_g \notin S$, le groupe ne contribue pas, même homogène.
- **Valeurs manquantes.** Toute valeur NA dans c invalide le test d'unanimité; le terme `cov_fullmatch` arrête l'exécution avec un message explicite (fail-fast).
- **Catégorie absente.** Si une `category=` ciblée n'est pas observée, l'initialiseur signale une erreur d'entrée avant la simulation.
- **Coût algorithmique.** À chaque toggle, seule la structure du groupe g concerné est mise à jour : tailles n_g , effectif dominant $n_{g,r\text{-max}}$, et état binaire F_g . L'actualisation est donc en $\mathcal{O}(1)$, ce qui rend l'effet compatible avec les grands réseaux bipartis.
- **Dégénérescence potentielle.** Si presque tous les groupes sont unanimement homogènes (ou tous hétérogènes), la statistique est quasi constante, entraînant des coefficients extrêmes ($\pm\infty$) — comportement normal pour des cas saturés.

Appel dans `erpm()`.

```

1 # Cas général : tous les groupes homogènes
2 erpm(partition ~ cov_fullmatch("dept"))
3
4 # Filtrage par taille
5 erpm(partition ~ cov_fullmatch("dept", size = 2:3))
6
7 # Variante ciblée : unanimité sur la modalité "A"
8 erpm(partition ~ cov_fullmatch("dept", category = "A", size =
  2:3))

```

8 Homophilie discrète par cliques : `cov_match(k)`

Principe.

`cov_match(k)` mesure l'homogénéité d'un attribut catégoriel c à *ordre de clique* fixé $k \geq 1$ au sein des groupes d'une partition. Il compte, pour chaque groupe, les

sous-ensembles de k acteurs partageant la même modalité.

Définition formelle.

Pour une partition stricte $B = (A, G, E)$ et $c = (c_i)$, on note $n_{g,r} = |\{i \in g : c_i = r\}|$. La statistique d'ordre k est

$$S_k(B; c) = \sum_{g \in G} \sum_{r \in L} \binom{n_{g,r}}{k},$$

Cas particuliers :

$$S_1(B; c) = N \quad \text{et} \quad S_2(B; c) = \sum_{i < j} \mathbf{1}[c_i = c_j] w_{ij},$$

où w_{ij} est le nombre de groupes communs à (i, j) . Variante ciblée (`category= κ`) :

$$S_k^{(\kappa)}(B; c) = \sum_{g \in G} \binom{n_{g,\kappa}}{k},$$

Lien avec la projection 1-mode.

Chaque groupe g induit une clique K_{n_g} entre ses membres. S_k compte, dans cette projection, les k -cliques *monochromatiques* de c . Contrairement à `nodematch`, le comptage est *pondéré par co-appartenance multiple* via w_{ij} et s'étend à $k > 2$.

Change-stat locale.

Un toggle (i, g) modifie uniquement le compteur de la modalité $r^* = c_i$ dans g . Si $m = n_{g,r^*}$ avant l'opération :

$$\Delta S_k = \begin{cases} \binom{m}{k-1}, & \text{ajout } (i \notin g \rightarrow i \in g), \\ -\binom{m-1}{k-1}, & \text{retrait } (i \in g \rightarrow i \notin g), \end{cases}$$

avec la convention $\binom{n}{t} = 0$ si $n < t$. Calcul en $\mathcal{O}(1)$.

Normalisations.

Trois variantes optionnelles :

$$S_k^{\text{none}} = \sum_{g,r} \binom{n_{g,r}}{k}, \quad S_k^{\text{by_group}} = \sum_g \frac{\sum_r \binom{n_{g,r}}{k}}{\binom{n_g}{k}}, \quad S_k^{\text{global}} = \frac{1}{\binom{N}{k}} \sum_{g,r} \binom{n_{g,r}}{k},$$

`by_group` est bornée dans $[0, \#\{g : n_g \geq k\}]$ et vaut le nombre de groupes non vides pour $k = 1$.

Interprétation.

- k petit \Rightarrow sensibilité à l'homogénéité locale (paires, triades).
- k grand \Rightarrow focalisation sur des groupes fortement homogènes.
- Additivité sur groupes et catégories, mise à jour locale stable.

Appel dans `erpm()`.

```
1 # Comptage des paires monochromatiques
2 erpm(partition ~ cov_match("dept", clique_size = 2))
3
4 # Cliques d'ordre 3, normalisation par groupe
5 erpm(partition ~ cov_match("dept", clique_size = 3, normalized
6   = "by_group"))
7
8 # Modalité ciblée
9 erpm(partition ~ cov_match("dept", clique_size = 2, category =
10   "A"))
```

Résumé.

`cov_match(k)` est une mesure discrète de l'homophilie intra-groupe à ordre fixe. Elle compte les k -cliques monochromatiques dans la projection des acteurs, admet des normalisations `by_group/global`, et se met à jour en $O(1)$ par toggle. Elle constitue la brique élémentaire de la version lissée `cov_match_GW(λ)`.

9 Homophilie géométriquement pondérée : `cov_match_GW(λ)`

Principe.

Extension continue de `cov_match`, l'effet `cov_match_GW(λ)` agrège les coïncidences de modalités au sein des groupes selon une **série géométrique à décroissance contrôlée par $\lambda > 1$** . Il mesure l'homophilie intra-groupe à tous les ordres de cliques simultanément.

Définition formelle.

Pour une partition stricte $B = (A, G, E)$ et un attribut catégoriel $c = (c_i)$, on note $n_{g,r} = |\{i \in g : c_i = r\}|$. La statistique s'écrit :

$$S_{\text{GW}}(B; c, \lambda) = \sum_{n \geq 1} \left(-\frac{1}{\lambda}\right)^{n-1} \sum_{g \in G} \sum_{r \in L} \binom{n_{g,r}}{n},$$

Chaque terme $\binom{n_{g,r}}{n}$ compte les cliques monochromatiques d'ordre n dans la projection 1-mode des acteurs.

Forme fermée.

En sommant sur n par application du binôme de Newton, on obtient :

$$S_{\text{GW}}(B; c, \lambda) = \sum_{g \in G} \sum_{r \in L} \lambda \left[1 - \left(\frac{\lambda-1}{\lambda}\right)^{n_{g,r}}\right],$$

Le terme $S(n_{g,r}, \lambda) = \lambda[1 - r_\lambda^{n_{g,r}}]$, avec $r_\lambda = (\lambda - 1)/\lambda \in [0, 1)$, représente la contribution d'une modalité r dans un groupe g .

Lien avec cov_match.

`cov_match(k)` correspond au cas discret d'ordre fixe k , avec

$$S_k(B; c) = \sum_{g,r} \binom{n_{g,r}}{k},$$

L'effet `cov_match_GW` en est la somme pondérée $\sum_{k \geq 1} (-1/\lambda)^{k-1} S_k$, ce qui lui confère une *continuité sur tous les ordres*. Il ne peut pas être reproduit dans `ergm` par une simple combinaison de `cov_match(k)` : il exigerait des poids géométriques fixes (offsets), dépendants de la taille maximale des groupes et instables numériquement.

Change-stat locale.

Un toggle (i, g) de modalité $r^* = c_i$ modifie uniquement le compteur n_{g,r^*} . Si $m = n_{g,r^*}$ avant l'opération :

$$\Delta S = \begin{cases} r_\lambda^m, & \text{ajout } (i \notin g \rightarrow i \in g), \\ -r_\lambda^{m-1}, & \text{retrait } (i \in g \rightarrow i \notin g), \end{cases}$$

Le calcul est en $O(1)$, indépendant de la taille du groupe.

Normalisations.

Trois variantes optionnelles :

$$\begin{aligned} S_{\text{GW}}^{\text{none}} &= \sum_{g,r} \lambda(1 - r_\lambda^{n_{g,r}}), \\ S_{\text{GW}}^{\text{by_group}} &= \sum_g \frac{\sum_r \lambda(1 - r_\lambda^{n_{g,r}})}{\lambda(1 - r_\lambda^{n_g})}, \\ S_{\text{GW}}^{\text{global}} &= \frac{\sum_{g,r} \lambda(1 - r_\lambda^{n_{g,r}})}{\lambda(1 - r_\lambda^N)}, \end{aligned}$$

Elles permettent de comparer l'homogénéité *intra-groupe* ou *globale* entre partitions de tailles différentes.

Interprétation.

- λ grand \Rightarrow pondération forte des petites cliques ($k = 1, 2$).
- λ proche de 1 \Rightarrow prise en compte accrue des grandes cliques.
- Additivité sur groupes et modalités, stabilité MCMC par décroissance géométrique.

Appel dans erpm().

```
1 # Forme de base
2 erpm(partition ~ cov_match_GW("dept", lambda = 2))
3
4 # Normalisation et modalité ciblée
5 erpm(partition ~ cov_match_GW("dept", lambda = 3,
6                               category = "A",
7                               normalized = "by_group"))
```

Résumé.

`cov_match_GW(λ)` généralise `cov_match` en intégrant tous les ordres de cliques homogènes dans une mesure unique et lissée d'homophilie intra-groupe. La forme fermée $\lambda[1 - r_\lambda^{n_{g,r}}]$ garantit un calcul local, additif et stable.

10 Effet cov_fulldiff

Cadre.

On partitionne des acteurs $A = \{1, \dots, N\}$ en groupes G . Le biparti associé est $B = (A, G, E)$. Chaque acteur i appartient à un unique groupe $g(i) \in G$. La taille d'un groupe g est $n_g = \deg_B(g) = |g|$.

Attribut et filtre de taille.

On fixe un attribut numérique d'acteur $x = (x_i)_{i \in A}$, par exemple un âge ou un score. Pour chaque groupe g , on note

$$x_g^{\min} = \min_{i \in g} x_i, \quad x_g^{\max} = \max_{i \in g} x_i, \quad D_g = x_g^{\max} - x_g^{\min},$$

Par convention, si $n_g \leq 1$ alors $D_g = 0$. On choisit un ensemble de tailles admissibles $S \subset \mathbb{N}_{\geq 1}$ (par exemple $S = \{k\}$, $S = \{a, \dots, b\}$, ou $S = \mathbb{N}_{\geq 2}$).

Définition.

La statistique `cov_fulldiff` somme les écarts maximaux intra-groupe sur les groupes dont la taille appartient à S :

$$T(B; x, S) = \sum_{g \in G} D_g \mathbf{1}[n_g \in S] = \sum_{g \in G} (x_g^{\max} - x_g^{\min}) \mathbf{1}[n_g \in S],$$

Par défaut on prend $S = \mathbb{N}_{\geq 2}$, de sorte que les singletons ne contribuent jamais.

Lien avec `absdiff`.

Sur la projection mode 1, un groupe g engendre un complet K_{n_g} . Le terme `absdiff` contribuerait, pour ce groupe,

$$T_{\text{absdiff},g} = \sum_{i < j, i, j \in g} |x_i - x_j|,$$

En général, $D_g = x_g^{\max} - x_g^{\min}$ ne peut pas être écrit comme une combinaison linéaire simple des $|x_i - x_j|$. Par exemple, pour $(x_1, x_2, x_3) = (0, 2, 2)$ on a $D_g = 2$ mais $T_{\text{absdiff},g} = 4$. Aucun facteur fixe ne permet de passer de l'un à l'autre pour tous les profils. `cov_fulldiff` ne peut donc pas être reproduit par un empilement d'effets de type `absdiff`.

Change-stat pour un déplacement $i : a \rightarrow b$.

On considère un déplacement d'un acteur i d'un groupe a vers un groupe b , avec tailles et bornes avant déplacement

$$n_a, n_b, \quad x_a^{\min}, x_a^{\max}, \quad x_b^{\min}, x_b^{\max},$$

Après déplacement, on obtient

$$n'_a = n_a - 1, \quad n'_b = n_b + 1,$$

et de nouvelles bornes $(x_a^{\min'}, x_a^{\max'})$, $(x_b^{\min'}, x_b^{\max'})$, d'où

$$D'_a = x_a^{\max'} - x_a^{\min'}, \quad D'_b = x_b^{\max'} - x_b^{\min'},$$

La variation de la statistique est

$$\Delta T = D'_b \mathbf{1}[n'_b \in S] - D_b \mathbf{1}[n_b \in S] + D'_a \mathbf{1}[n'_a \in S] - D_a \mathbf{1}[n_a \in S],$$

Les autres groupes sont inchangés. Le calcul est local : il suffit de maintenir, pour chaque groupe, ses bornes (x_g^{\min}, x_g^{\max}) et leur mise à jour lors des déplacements.

Lecture.

`cov_fulldiff` mesure la dispersion interne maximale des groupes sur une échelle numérique, en ne tenant compte que des tailles dans S . Des groupes nombreux et très étalés sur x augmentent fortement la statistique ; des groupes homogènes (valeurs quasi constantes) contribuent peu ou pas. :contentReference[oaicite :0]index=0

11 Dispersion intra-groupe sur attribut numérique : `cov_diff`

Principe.

L'effet `cov_diff` mesure l'hétérogénéité d'un attribut monadique numérique au sein des groupes d'une partition. Il agrège, pour chaque groupe, la *différence maximale* de l'attribut dans toutes les k -cliques d'acteurs co-groupés (`clique_size = k`).

Définition formelle.

On considère une partition stricte d'acteurs $A = \{1, \dots, N\}$ en groupes G , et un attribut numérique $x = (x_i)_{i \in A} \in \mathbb{R}^N$. Pour un groupe $g \in G$, on note $\mathcal{C}_k(g)$ l'ensemble des sous-ensembles $S \subset g$ de taille $|S| = k$ (avec $\mathcal{C}_k(g) = \emptyset$ si $n_g < k$). La différence maximale dans une clique est

$$D(S) = \max_{i \in S} x_i - \min_{i \in S} x_i,$$

La statistique non normalisée est alors

$$T_k(B; x) = \sum_{g \in G} \sum_{S \in \mathcal{C}_k(g)} D(S),$$

Cas particulier $k = 2$ et lien avec **absdiff**.

Lorsque $k = 2$, chaque clique se réduit à une paire $\{i, j\}$ et l'on obtient

$$T_2(B; x) = \sum_{g \in G} \sum_{\substack{i < j \\ i, j \in g}} |x_i - x_j|,$$

En introduisant la fonction indicatrice $\mathbf{1}[\gamma(i) = \gamma(j)]$, où $\gamma(i)$ désigne le groupe unique de l'acteur i , on peut réécrire

$$T_2(B; x) = \sum_{i < j} \mathbf{1}[\gamma(i) = \gamma(j)] |x_i - x_j|,$$

Sur la projection mode 1 simple $\tilde{y}_{ij} = \mathbf{1}[\gamma(i) = \gamma(j)]$, l'effet **absdiff(cov)** de **ergm** calcule exactement la même quantité. Ainsi, pour $k = 2$ et une partition stricte, **cov_diff** non normalisé coïncide avec **absdiff(cov)** sur la projection.

Normalisation.

Pour limiter l'effet mécanique de la taille des groupes, une normalisation « par groupe » remplace la somme brute par une moyenne de dispersion par clique :

$$T_k^{\text{by-group}}(B; x) = \sum_{g \in G} \frac{1}{\binom{n_g}{k}} \sum_{S \in \mathcal{C}_k(g)} D(S),$$

où $\binom{n_g}{k}$ est le nombre de k -cliques possibles dans g . Chaque groupe contribue alors en fonction de son niveau d'hétérogénéité interne, et non plus seulement de sa taille.

Change-stat locale.

Dans l'ERGM biparti, un toggle (i, g) (ajout ou retrait de l'acteur i dans le groupe g) ne modifie que les cliques internes à ce groupe. Pour $k = 2$, la variation s'écrit simplement

$$\Delta T_2 = \begin{cases} \sum_{j \in g^-} |x_i - x_j| & (\text{ajout de } (i, g)), \\ - \sum_{j \in g^+ \setminus \{i\}} |x_i - x_j| & (\text{retrait de } (i, g)), \end{cases}$$

avec g^- et g^+ les contenus du groupe avant/après toggle. Pour $k > 2$, seules les cliques de taille k contenant i sont affectées, et la change-statistic reste locale au groupe g . Cette structure locale permet un calcul peu coûteux dans les chaînes MCMC d'estimation ERGM, en particulier pour $k = 2$.

Appel dans **erpm()**.

```
1 # Dispersion dyadique intra-groupe (k = 2)
2 erpm(partition ~ cov_diff("age", clique_size = 2))
3
4 # Variante normalisée par groupe
5 erpm(partition ~ cov_diff("age", clique_size = 2,
6                           normalized = TRUE))
```

Résumé.

cov_diff fournit une mesure d'hétérogénéité intra-groupe adaptée aux attributs numériques. Pour $k = 2$, il correspond à la somme des différences absolues entre acteurs co-groupés (et coïncide avec **absdiff** sur la projection). Pour $k > 2$, il généralise cette logique à des cliques de taille k , sans équivalent dyadique direct dans **ergm**, tout en restant compatible avec une implémentation locale efficace côté change-statistic.

12 Dispersion multi-échelle sur attribut numérique : cov_diff_GW

Principe.

L'effet `cov_diff_GW` étend `cov_diff` en combinant, dans une même statistique, l'hétérogénéité d'un attribut numérique à plusieurs tailles de cliques $k = 2, 3, \dots$. Pour chaque taille de clique, on calcule une statistique de type `cov_diff` puis on agrège ces contributions via des poids géométriques alternés $(-1/\lambda)^{k-1}$, contrôlés par un paramètre $\lambda > 1$.

Définition formelle.

On considère une partition stricte d'acteurs $A = \{1, \dots, N\}$ en groupes G , et un attribut numérique $x = (x_i)_{i \in A} \in \mathbb{R}^N$. Pour un groupe $g \in G$, on note $\mathcal{C}_k(g)$ l'ensemble des sous-ensembles $S \subset g$ de taille $|S| = k$ (avec $\mathcal{C}_k(g) = \emptyset$ si $n_g < k$).

La différence maximale dans une clique est

$$D(S) = \max_{i \in S} x_i - \min_{i \in S} x_i,$$

Pour chaque taille $k \geq 2$, on définit la statistique *de base* de type `cov_diff`

$$c_k(\text{cov}, p) = \sum_{g \in G} \sum_{S \in \mathcal{C}_k(g)} D(S),$$

qui généralise $T_k(B; x)$ à la famille $(c_k)_{k \geq 2}$. L'effet `cov_diff_GW(cov, lambda)` est alors donné par la combinaison géométrique

$$T_{\text{GW}}(p; x; \lambda) = \sum_{k=2}^{K_{\max}} \left(-\frac{1}{\lambda} \right)^{k-1} c_k(\text{cov}, p), \quad \lambda > 1,$$

où $K_{\max} = \max_{g \in G} n_g$ est la taille maximale de groupe. Les termes de grande taille k sont automatiquement amortis par les facteurs $\lambda^{-(k-1)}$.

Lien avec cov_diff et cas particulier $k = 2$.

Pour une taille de clique fixée k , la statistique $c_k(\text{cov}, p)$ coïncide avec la `cov_diff` non normalisée de taille k :

$$c_k(\text{cov}, p) \equiv T_k(B; x),$$

En particulier, pour $k = 2$,

$$c_2(\text{cov}, p) = \sum_{g \in G} \sum_{\substack{i < j \\ i, j \in g}} |x_i - x_j| = \sum_{i < j} \mathbf{1}[\gamma(i) = \gamma(j)] |x_i - x_j|,$$

qui est la somme des différences absolues entre paires d'acteurs co-groupés. L'effet `cov_diff_GW` contient donc, comme première composante, la mesure dyadique $c_2(\text{cov}, p)$, puis ajoute des corrections pondérées associées aux cliques de taille $k \geq 3$.

Interprétation.

La série c_2, c_3, \dots décrit comment l'attribut se disperse dans des sous-ensembles de plus en plus grands au sein de chaque groupe. Les poids $(-1/\lambda)^{k-1}$ permettent :

- de contrôler la contribution des grandes cliques via λ (décroissance géométrique),
- d'alterner le signe des contributions (tailles paires vs impaires), ce qui introduit un effet de compensation entre niveaux de taille.

On obtient ainsi une mesure d'hétérogénéité « multi-échelle » intra-groupe, sensiblement plus riche qu'un effet purement dyadique.

Change-stat locale.

Dans l'ERGM biparti, un toggle (i, g) (ajout ou retrait de l'acteur i dans le groupe g) ne modifie que les cliques internes à ce groupe. Pour chaque taille k , la variation locale est

$$\Delta c_k = \sum_{S \in \mathcal{C}_k(g^+)} D(S) - \sum_{S \in \mathcal{C}_k(g^-)} D(S),$$

où g^- et g^+ sont les contenus du groupe avant/après toggle. La variation de `cov_diff_GW` s'obtient ensuite par

$$\Delta T_{\text{GW}} = \sum_{k=2}^{K_{\text{max}}} \left(-\frac{1}{\lambda}\right)^{k-1} \Delta c_k,$$

Un schéma efficace consiste, pour chaque toggle, à accumuler d'abord les Δc_k dans un petit vecteur $(\Delta c_k)_{k \geq 2}$, puis à appliquer une fois pour toutes les poids géométriques (pré-calculés) dans la combinaison linéaire ci-dessus. La change-statistic reste ainsi strictement locale au groupe g , avec un surcoût contrôlé par le nombre de tailles de cliques considérées.

Appel dans `erpm()`.

```
1 # Version par défaut (lambda = 2)
2 erpm(partition ~ cov_diff_GW("age"))
3
4 # Variante avec poids plus courts (lambda plus grand)
5 erpm(partition ~ cov_diff_GW("age", lambda = 3))
```

Résumé.

`cov_diff_GW` fournit une mesure d'hétérogénéité intra-groupe multi-échelle pour un attribut numérique. Il s'appuie sur les $c_k(\text{cov}, p)$ des statistiques `cov_diff` de taille k , puis les combine via des poids géométriques $(-1/\lambda)^{k-1}$. Pour $k = 2$, la composante c_2 retrouve la somme des différences absolues entre acteurs co-groupés. Pour $k \geq 3$, l'effet devient sensible à la dispersion de l'attribut dans des cliques plus larges, sans équivalent direct dans les effets dyadiques standard de `ergm`, tout en restant compatible avec une implémentation locale efficace côté change-statistic.

13 Covariée dyadique intra-groupe : `dyadcov_full`

Principe.

L'effet `dyadcov_full` agrège une covariée dyadique numérique définie entre acteurs, en ne retenant que les paires d'acteurs appartenant au même groupe d'une

partition. Il combine donc trois objets : une partition stricte des acteurs, son graphe biparti ERPM, et une matrice dyadique $Z = (z_{ij})$ définie sur les paires d'acteurs.

Définition formelle.

On considère une partition stricte d'acteurs $A = \{1, \dots, N\}$ en groupes G , avec $\gamma(i)$ le groupe de l'acteur i , et $A(g) = \{i \in A : \gamma(i) = g\}$. On fixe une covariée dyadique réelle $Z = (z_{ij})_{1 \leq i, j \leq N}$ telle que : $z_{ij} = z_{ji}$, $z_{ii} = 0$, et pour $i \neq j$, $z_{ij} \in \mathbb{R}$ mesure une caractéristique de la dyade (i, j) (amitié, distance, similarité, etc.).

Pour chaque groupe g , on définit la contribution locale :

$$S_g(Z) = \sum_{\substack{i < j \\ i, j \in A(g)}} z_{ij},$$

c'est-à-dire la somme des covariées dyadiques sur toutes les paires d'acteurs internes à g .

On ajoute un filtre optionnel sur la taille des groupes défini par un ensemble $S \subset \mathbb{N}$ et une fonction indicatrice

$$\phi_{\text{size}}(n_g; S) = \begin{cases} 1, & \text{si } S = \emptyset, \\ 1, & \text{si } S \neq \emptyset \text{ et } n_g \in S, \\ 0, & \text{si } S \neq \emptyset \text{ et } n_g \notin S, \end{cases}$$

La statistique globale est alors

$$T_{\text{full}}(p; Z, S) = \sum_{g \in G} \phi_{\text{size}}(n_g; S) \sum_{\substack{i < j \\ i, j \in A(g)}} z_{ij},$$

Cas particulier binaire.

Si la covariée dyadique est binaire, $z_{ij} \in \{0, 1\}$, alors $S_g(Z)$ compte simplement le nombre de dyades (i, j) actives au sein du groupe g . La statistique $T_{\text{full}}(p; Z, S)$ correspond alors au nombre total de dyades « actives » entre acteurs co-groupés, éventuellement restreint aux groupes dont la taille appartient à S .

Lien avec edgescov.

En absence de filtre (`size = NULL`), on peut réécrire :

$$T_{\text{full}}(p; Z, \emptyset) = \sum_{g \in G} \sum_{\substack{i < j \\ i, j \in A(g)}} z_{ij} = \sum_{i < j} \mathbf{1}[\gamma(i) = \gamma(j)] z_{ij},$$

Sur la projection mode 1, \tilde{Y} définie par $\tilde{y}_{ij} = \mathbf{1}[\gamma(i) = \gamma(j)]$, le terme `edgescov(Z)` de `ergm` calcule exactement $\sum_{i < j} \tilde{y}_{ij} z_{ij}$. Ainsi, pour une partition stricte et `size = NULL`, `dyadcov_full` coïncide avec `edgescov` appliqué à la matrice dyadique sur le graphe projeté.

Dès que l'on impose un filtre de taille non trivial (`size ≠ NULL`), le poids $\phi_{\text{size}}(n_g; S)$ dépend explicitement de n_g et la statistique ne se réduit plus à un unique `edgescov` standard.

Deux stratégies sont envisageables pour intégrer l'effet `dyadcov_full` dans la chaîne ERPM → ERGM.

(1) Implémentation native : `InitErgmTerm` + fichier C.

On définit un initialiseur R dédié qui prépare les entrées (partition, filtre de taille, matrice dyadique) et une change-statistic en C qui recalcule localement, pour chaque toggle (i, g) , la variation de la somme $\sum_{j \in A(g)} z_{ij}$. Cette approche respecte la logique bipartite et permet une évaluation locale en $O(n_g)$ par toggle.

(2) Traduction via le `erpm_wrapper`.

Une autre possibilité serait de traduire `dyadcov_full` en un `edgescov` classique appliqué à une matrice dyadique restreinte aux acteurs appartenant aux groupes autorisés. Le filtre `size` pourrait être simulé en reconstruisant, pour chaque état, une matrice dyadique réduite où l'on supprime les lignes et colonnes des acteurs situés dans des groupes de taille non admissible. Cette approche reste théoriquement cohérente mais conduit à une matrice dyadique dépendante de la partition, donc du graphe biparti courant. Cela viole l'hypothèse standard d'ERGMs selon laquelle les covariées dyadiques sont fixées et ne dépendent pas de l'état de la chaîne.

Choix retenu.

Nous adoptons la première stratégie (construction d'un `InitErgmTerm` et d'un fichier C) pour deux raisons principales :

- elle garantit une change-statistic locale, stable et indépendante des covariées externes, ce qui est essentiel pour l'efficacité de la chaîne MCMC ;
- elle respecte strictement la sémantique ERGM : les covariées dyadiques demeurent fixes, seul le graphe biparti évolue.

Change-stat locale.

Dans l'ERGM, la partition est représentée par un graphe biparti $B = (A, G, E)$ où une arête (i, g) code l'appartenance de l'acteur i au groupe g . Un toggle (i, g) (ajout ou retrait) modifie uniquement :

- la taille n_g ,
- l'ensemble des dyades (i, j) internes à g sur lesquelles $S_g(Z)$ est calculé.

La matrice Z reste fixe. La variation de la statistique globale est

$$\Delta T_{\text{full}} = \phi_{\text{size}}(n_g^+; S) S_g^+ - \phi_{\text{size}}(n_g^-; S) S_g^-,$$

où $(\cdot)^-$ et $(\cdot)^+$ désignent respectivement l'état avant et après toggle. En pratique, l'implémentation C recalcule localement la somme $\sum_j z_{ij}$ sur les membres du groupe, ainsi que $\phi_{\text{size}}(n_g^-; S)$ et $\phi_{\text{size}}(n_g^+; S)$, puis combine ces termes selon qu'il s'agit d'un ajout ou d'un retrait.

Appel dans `erpm()`.

```
1 ## Covariée dyadique intra-groupe sans filtre de taille
2 erpm(partition ~ dyadcov_full("friendship"))
3
4 ## Variante avec filtre sur les tailles (par exemple groupes
5   de taille 3 ou 4)
6 erpm(partition ~ dyadcov_full("friendship", size = c(3, 4)))
```

Résumé.

`dyadcov_full` agrège une covariée dyadique définie entre acteurs, en ne retenant que les dyades internes aux groupes d’une partition. La statistique est une somme, éventuellement filtrée par taille de groupe, des $S_g(Z)$ calculés sur chaque groupe. Lorsque `size` est absent, l’effet se réduit à un `edgescov` sur le graphe projeté où les arêtes codent la co-appartenance à un groupe. La change-statistic reste locale à un seul groupe à chaque toggle (i, g) , ce qui permet une implémentation efficace dans les chaînes MCMC d’estimation ERGM.

14 Covariée dyadique sur cliques intra-groupe : `dyadcov`

Principe.

L’effet `dyadcov` généralise `dyadcov_full` et `cliques(k)`. Il agrège une covariée dyadique numérique définie entre acteurs, mais non plus seulement sur les paires d’acteurs d’un même groupe : il la combine sur toutes les *cliques* de taille fixée k au sein de chaque groupe. Un paramètre `normalized` permet en option de normaliser par le nombre théorique de cliques par groupe.

Définition formelle.

On considère une partition stricte d’acteurs $A = \{1, \dots, N\}$ en groupes G , avec $\gamma(i)$ le groupe de l’acteur i , et $A(g) = \{i \in A : \gamma(i) = g\}$, de taille $n_g = |A(g)|$. On fixe une covariée dyadique réelle $Z = (z_{ij})_{1 \leq i, j \leq N}$ telle que : $z_{ij} = z_{ji}$, $z_{ii} = 0$, et, pour $i \neq j$, $z_{ij} \in \mathbb{R}$ mesure une caractéristique de la dyade (i, j) (intensité d’interaction, similarité, distance, etc.).

Pour une taille de clique `clique_size` = $k \geq 2$ et un groupe g , on note

$$\mathcal{C}_k(g) = \{ C \subseteq A(g) : |C| = k \}$$

l’ensemble des cliques d’acteurs de taille k dans g . Par construction, leur nombre est

$$|\mathcal{C}_k(g)| = \binom{n_g}{k},$$

de sorte que l’effet de base `cliques(k)` correspond à la statistique pure de comptage

$$C_k(p) = \sum_{g \in G} |\mathcal{C}_k(g)| = \sum_{g \in G} \binom{n_g}{k},$$

L’effet `dyadcov` reprend ces mêmes cliques, mais en leur associant un *poids dyadique*. Pour une clique $C = \{i_1, \dots, i_k\}$, on considère toutes les dyades (i_a, i_b) avec $1 \leq a < b \leq k$, au nombre de $\binom{k}{2}$, et on définit le poids

$$P(C; Z) = \prod_{\substack{i < j \\ i, j \in C}} z_{ij},$$

La contribution locale du groupe g est alors

$$S_g^{(k)}(Z) = \sum_{C \in \mathcal{C}_k(g)} P(C; Z) = \sum_{C \in \mathcal{C}_k(g)} \prod_{\substack{i < j \\ i, j \in C}} z_{ij},$$

Statistique globale non normalisée.

La statistique globale associée à `dyadcov(dyadcov, clique_size = k, normalized = FALSE)` est

$$T^{(k)}(p; Z) = \sum_{g \in G} S_g^{(k)}(Z) = \sum_{g \in G} \sum_{C \in \mathcal{C}_k(g)} \prod_{\substack{i < j \\ i, j \in C}} z_{ij},$$

Cas particulier par défaut : $k = 2$.

Lorsque `clique_size` = 2, une “clique” C est simplement une dyade $\{i, j\}$ (avec $i \neq j$). On a alors

$$P(\{i, j\}; Z) = z_{ij},$$

et les cliques de taille 2 dans un groupe g sont exactement les paires (i, j) d'acteurs du groupe. On obtient

$$S_g^{(2)}(Z) = \sum_{\substack{i < j \\ i, j \in A(g)}} z_{ij}, \quad T^{(2)}(p; Z) = \sum_{g \in G} \sum_{\substack{i < j \\ i, j \in A(g)}} z_{ij},$$

ce qui coïncide avec la statistique de l'effet `dyadcov_full(dyadcov, size = NULL)`. Comme pour `dyadcov_full`, on peut réécrire

$$T^{(2)}(p; Z) = \sum_{i < j} \mathbf{1}[\gamma(i) = \gamma(j)] z_{ij},$$

et donc l'effet `dyadcov(dyadcov, clique_size=2, normalized=FALSE)` coïncide avec `edgecov(dyadcov)` sur le graphe projeté \tilde{Y} dont les arêtes codent la co-appartenance à un groupe.

Version normalisée.

Pour réduire l'effet des groupes très grands, une version normalisée par taille est proposée via `normalized = TRUE`. Le nombre de cliques potentielles de taille k dans un groupe g est

$$D_g^{(k)} = \binom{n_g}{k} = \frac{n_g!}{k!(n_g - k)!},$$

On définit alors la contribution normalisée

$$\bar{S}_g^{(k)}(Z) = \begin{cases} \frac{1}{\binom{n_g}{k}} S_g^{(k)}(Z), & \text{si } n_g \geq k, \\ 0, & \text{si } n_g < k, \end{cases}$$

et la statistique globale normalisée

$$T_{\text{norm}}^{(k)}(p; Z) = \sum_{g \in G} \bar{S}_g^{(k)}(Z) = \sum_{g \in G} \mathbf{1}[n_g \geq k] \frac{1}{\binom{n_g}{k}} \sum_{C \in \mathcal{C}_k(g)} \prod_{\substack{i < j \\ i, j \in C}} z_{ij},$$

Pour $k = 2$, $\bar{S}_g^{(2)}(Z)$ est simplement la moyenne de la covariée dyadique sur toutes les paires d'acteurs du groupe g .

Change-stat locale.

Comme pour `dyadcov_full`, la partition est représentée par un graphe biparti $B = (A, G, E)$ où une arête (i, g) code l'appartenance de l'acteur i au groupe g . Un toggle (i, g) (ajout ou retrait) modifie uniquement :

- la taille n_g ,
- l'ensemble des cliques $\mathcal{C}_k(g)$ qui contiennent i ,
- et les contributions associées $S_g^{(k)}(Z)$ (et éventuellement leur normalisation).

La matrice Z reste fixe. La variation de la statistique non normalisée s'écrit

$$\Delta T^{(k)} = T^{(k)+} - T^{(k)-} = S_g^{(k)+} - S_g^{(k)-},$$

où l'on ajoute (respectivement soustrait) les poids $P(C; Z)$ des cliques créées (respectivement détruites) par le toggle. Pour la version normalisée, on remplace $S_g^{(k)}$ par $\bar{S}_g^{(k)}$, ce qui ne change la change-statistic qu'au niveau du groupe g .

Appel dans `erpm()`.

```
1 ## Produit des covariées dyadiques sur toutes les paires intra
   -groupe (k = 2)
2 erpm(partition ~ dyadcov("friendship", clique_size = 2,
   normalized = FALSE))
3
4 ## Version normalisée par groupe (moyenne sur les cliques)
   pour k = 3
5 erpm(partition ~ dyadcov("friendship", clique_size = 3,
   normalized = TRUE))
```

Résumé.

`dyadcov` est une extension de `dyadcov_full` et de `cliques(k)` : il agrège une covariée dyadique sur les cliques d'acteurs situées dans un même groupe, en offrant une version brute (somme des poids) et une version normalisée (moyenne par groupe). Pour `clique_size = 2` et `normalized = FALSE`, il se réduit exactement

à `dyadcov_full` et à un `edgescov` sur le graphe projeté des co-appartenances. La change-statistic reste locale à un seul groupe à chaque toggle (i, g) , ce qui permet une implémentation efficace pour l'estimation ERGM.

Annexe — ERGM et Metropolis–Hastings

A.1 Rappel : qu’est-ce qu’un ERGM ?

Un modèle de graphe aléatoire exponentiel (ERGM, *Exponential Random Graph Model*) définit une loi de probabilité sur l’ensemble des graphes possibles y :

$$\Pr_{\theta}(Y = y) \propto \exp(\theta^{\top} g(y)),$$

où :

- $g(y)$ est le **vecteur de statistiques du graphe** : il regroupe des caractéristiques mesurées sur le réseau, par exemple le *nombre d’arêtes* (statistique dénombrable ou « brute ») et le *degré moyen* (statistique agrégée ou « moyenne »).
- θ est le **vecteur de paramètres du modèle** : chaque composante pondère la propension du réseau à présenter la structure correspondante (par ex. triangles, réciprocité).
- \Pr_{θ} désigne la **distribution de probabilité induite par θ** : c’est la loi selon laquelle un graphe est susceptible d’être généré.

Motivation.

On dispose d’un **graphe observé** y_{obs} (données empiriques). Ce graphe est fixe : on calcule ses statistiques $g(y_{\text{obs}})$ une fois pour toutes. L’objectif est de trouver θ tel que les graphes simulés depuis \Pr_{θ} aient, *en moyenne*, les mêmes statistiques que y_{obs} :

$$\mathbb{E}_{\theta}[g(Y)] \approx g(y_{\text{obs}}),$$

afin de reproduire les régularités structurelles du réseau (densité, distribution des degrés, triangles, homophilie, etc.).

Principe général.

L’estimation de θ se fait par deux boucles :

- **Boucle interne (MCMC–MH)** : pour un θ fixé, on échantillonne des graphes selon \Pr_{θ} à l’aide de **Metropolis–Hastings (MH)**. Concrètement :
 1. partir d’un graphe initial $y^{(0)}$;
 2. proposer une modification locale donnant un candidat y' (ajout/suppression d’une arête) ;
 3. calculer la variation de log-probabilité ² :

$$\Delta = \theta^{\top} [g(y') - g(y)],$$

4. accepter y' avec probabilité $\alpha = \min(1, \exp(\Delta) \times \frac{q(y \rightarrow y')}{q(y' \rightarrow y)})$, où q est la loi de proposition.

En répétant ces étapes, on obtient une chaîne de Markov dont la loi stationnaire est précisément \Pr_{θ} . L’échantillon $\{y^{(s)}\}$, constitué de plusieurs graphes simulés indépendamment (après burn-in et thinning), sert à approximer les moyennes des statistiques $g(y)$ sous la distribution \Pr_{θ} .

- **Boucle externe (mise à jour de θ)** : après simulation, on compare la moyenne simulée $\mathbb{E}_{\theta^{(t)}}[g(Y)]$ à $g(y_{\text{obs}})$ et on ajuste

$$\theta^{(t+1)} = \theta^{(t)} + \alpha \left(g(y_{\text{obs}}) - \frac{1}{S} \sum_{s=1}^S g(y^{(s)}) \right),$$

où α est un **pas d’apprentissage**. Un pas trop grand peut faire diverger l’algorithme ; trop petit, la convergence devient lente.

2. Pour deux graphes y et y' sous un même θ :

$$\frac{\Pr_{\theta}(Y = y')}{\Pr_{\theta}(Y = y)} = \exp\left\{ \theta^{\top} [g(y') - g(y)] \right\} = \exp(\Delta), \quad \Delta = \theta^{\top} [g(y') - g(y)],$$

Si $\Delta > 0$, y' est plus probable que y (toutes choses égales par ailleurs).

Lien avec les algorithmes de gradient.

La mise à jour de θ est analogue à une **montée de gradient** sur la log-vraisemblance

$$\log L(\theta) = \theta^\top g(y_{\text{obs}}) - \log Z(\theta),$$

car $g(y_{\text{obs}}) - \mathbb{E}_\theta[g(Y)]$ joue le rôle d'un **gradient stochastique**. Le pas α influence vitesse et stabilité de convergence. La fonction de vraisemblance peut être **non convexe** dans l'espace des graphes : la convergence dépend du point de départ et de la structure du réseau.

A.2 Log-vraisemblance, déviance, AIC et BIC

Constante de normalisation $Z(\theta)$.

Dans un ERGM, l'écriture exacte de la probabilité de tirage est :

$$\Pr_\theta(Y = y) = \frac{\exp\{\theta^\top g(y)\}}{Z(\theta)}, \quad Z(\theta) = \sum_{y' \in \mathcal{Y}} \exp\{\theta^\top g(y')\},$$

où \mathcal{Y} est l'ensemble de *tous* les graphes possibles sur le même jeu de nœuds. $Z(\theta)$ **assure la normalisation** (les probabilités somment à 1). Comme \mathcal{Y} est gigantesque, $Z(\theta)$ n'est pas calculable exactement ; on l'approxime par MCMC.

Log-vraisemblance.

Idée générale.

La **vraisemblance** $L(\theta)$ mesure, pour un θ donné, à quel point le modèle juge plausible le graphe observé y_{obs} :

$$L(\theta) = \Pr_\theta(Y = y_{\text{obs}}) = \frac{\exp\{\theta^\top g(y_{\text{obs}})\}}{Z(\theta)},$$

Maximiser $L(\theta)$ (ou son logarithme) revient à choisir les paramètres qui rendent y_{obs} le plus probable dans la famille des modèles.

Pourquoi travailler au log.

On utilise la **log-vraisemblance**

$$\ell(\theta) = \log L(\theta) = \theta^\top g(y_{\text{obs}}) - \log Z(\theta),$$

car (i) les produits de probabilités deviennent des *sommes* (plus stables numériquement), et (ii) le calcul des dérivées/gradients est direct.

Déviance et critères d'information.

La **déviance** d'un modèle est définie par

$$D = -2 \ell(\hat{\theta}),$$

où $\hat{\theta}$ est l'estimateur (par exemple le maximum de vraisemblance). Sous conditions régulières, les **rapports de vraisemblance** ont une loi asymptotique de type χ^2 , d'où le facteur -2 qui permet d'interpréter les *différences* de déviance dans des tests de comparaison de modèles.

Déviance et AIC/BIC (version unifiée).

Dans un ERGM, $D = -2 \ell(\hat{\theta})$ sert d'indicateur global d'ajustement : plus $\ell(\hat{\theta})$ est grand, plus D est petit. La déviance mesure toutefois l'ajustement *pur* : un modèle plus complexe a presque toujours une déviance plus faible. Pour **équilibrer ajustement et complexité**, on utilise les critères d'information AIC/BIC :

$$\text{AIC} = 2k - 2 \ell(\hat{\theta}), \quad \text{BIC} = k \log n - 2 \ell(\hat{\theta}),$$

où k est le nombre de paramètres et n une taille effective (en réseau, souvent proche du nombre de dyades potentielles). On compare des modèles ajustés sur les mêmes données et on retient en pratique celui dont l'AIC/BIC est **minimal** (meilleur compromis « fidélité/parcimonie »). AIC/BIC permettent en outre de comparer des modèles *non nécessairement emboîtés*, ce que ne permet pas un test basé uniquement sur la différence de déviance.

A.3 Sorties usuelles de `summary(ergm)`

- **Call** : l'appel exact (traçabilité, reproductibilité).
- **Coefficients** : estimés $\hat{\theta}_i$ (un par statistique), **erreur-type** (SE), **statistique** z ($z = \hat{\theta}_i / \text{SE}(\hat{\theta}_i)$) et **p-valeur bilatérale** pour $H_0 : \theta_i = 0$. Un grand $|z|$ suggère un effet éloigné de 0 au regard de l'incertitude.
- **MCMC %** : fraction (en %) de l'erreur-type due au *bruit Monte-Carlo*. *Interprétation et actions* : un MCMC% élevé indique que l'échantillon simulé est peu informatif (forte autocorrélation). **Burn-in** = nombre d'itérations initiales *jetées* avant d'enregistrer des échantillons (laisser la chaîne se stabiliser); **Thinning** = ne garder qu'une itération sur m pour **décorrélérer les échantillons** entre eux (réduire l'autocorrélation sérielle); **Taille d'échantillon** = nombre total d'états conservés pour estimer les moyennes. Si la chaîne *mélange mal*, il peut être nécessaire d'ajuster les contraintes ou le mécanisme de proposition.
- **Log-vraisemblance, déviance D , AIC, BIC** : indicateurs globaux (définitions ci-dessus). AIC/BIC plus petits \Rightarrow meilleur compromis.

A.4 Contraintes et mécanisme de proposition (MH) en biparti

Contraintes (restriction de l'espace d'état).

Les **contraintes** (par exemple `b1part`) restreignent l'espace des graphes explorés par MH aux configurations *valides* : en biparti, interdire les arêtes intra-mode, imposer certaines bornes de degrés, etc. Elles ne constituent pas un « assouplissement » : elles **excluent** simplement les états invalides et garantissent que chaque graphe visité respecte la structure voulue.

Mécanisme de proposition.

Le **mécanisme de proposition** (souvent appelé « proposal ») génère un candidat y' à partir de y (par exemple tirer une dyade admissible et toggler l'arête), en *respectant* les contraintes. Il définit la distribution $q(y \rightarrow y')$ qui intervient dans

le **ratio de Hastings**. Si q est asymétrique (par exemple plus de façons de détruire que de créer une structure), le terme $\frac{q(y' \rightarrow y)}{q(y \rightarrow y')}$ **corrige** cette asymétrie dans la probabilité d'acceptation.

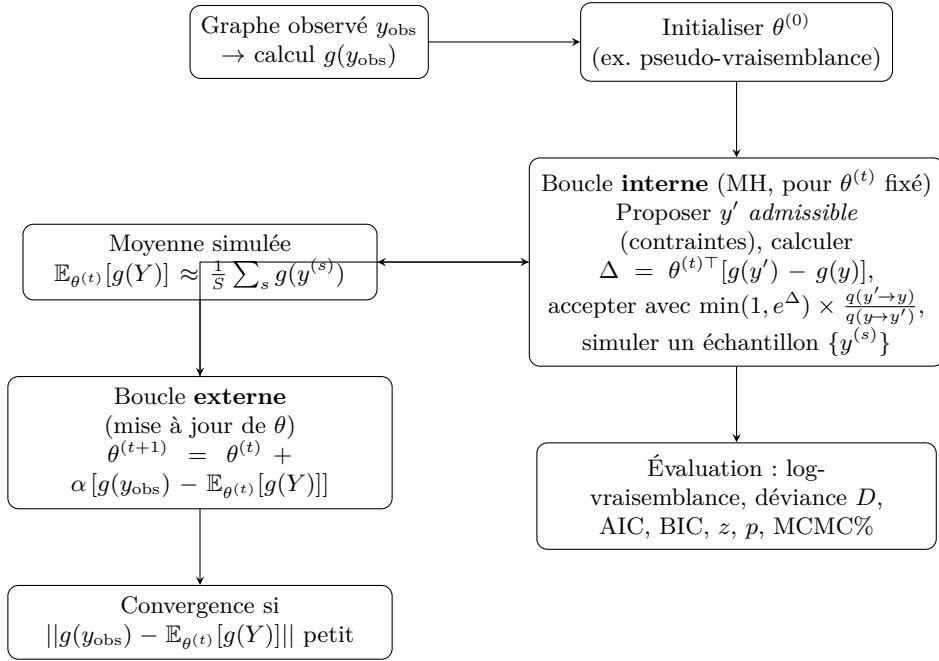
Pourquoi ces corrections sont indispensables.

Le couple « Δ du modèle » + « ratio de Hastings » assure la **réversibilité** (détail de balance) de la chaîne et garantit que la **loi stationnaire** visée est bien Pr_θ . Sans cette correction, la chaîne convergerait vers une distribution biaisée (dépendante du mécanisme de proposition) au lieu de la loi ERGM recherchée.

A.5 Exemple minimal (R)

```
1 library(ergm)
2 # Réseau biparti jouet : 4 acteurs -> 2 groupes
3 m <- matrix(c(1,0,
4               1,0,
5               0,1,
6               0,1), nrow=4, byrow=TRUE)
7 nw <- network::network(m, bipartite=2, directed=FALSE)
8
9 # Modèle 1 : densité (edges)
10 fit1 <- ergm(nw ~ edges)
11
12 # Modèle 2 : densité + distribution de degrés côté groupes (
13   mode 2)
14 fit2 <- ergm(nw ~ edges + b2degrange(from=2, to=3))
15
16 summary(fit1)
17 summary(fit2) # comparer log-vraisemblance, AIC, BIC, z, p-
18               values, MCMC%
```

A.6 Schéma du processus d'estimation



Deux boucles imbriquées : **interne** (MH : échantillonnage pour θ fixé) et **externe** (mise à jour de θ jusqu'à convergence).

A.7 Points d'attention

Choix de $g(y)$.

Les statistiques doivent refléter des mécanismes plausibles (densité, degrés, triangles, homophilie, effets d'attributs, etc.) sans surcharger le modèle : empiler trop de termes peut entraîner instabilités, colinéarités ou non-identifiabilité. Une approche progressive (noyau simple, diagnostics, puis complexification) est souvent utile.

Diagnostics MCMC.

Surveiller l'autocorrélation et la stabilité des moyennes simulées. Un **MCMC%** élevé signale une forte dépendance entre échantillons. Augmenter le **burn-in** (laisser la chaîne se stabiliser avant de collecter), appliquer du **thinning** (garder une itération sur m pour **décorrélérer les échantillons entre eux**), et/ou accroître la **taille d'échantillon**. Si le mélange reste faible, ajuster contraintes et mécanisme de proposition peut aider.

Comparaison de modèles.

AIC/BIC comparent des modèles ajustés sur les mêmes données en pénalisant la complexité. L'information est dans les **écarts** d'AIC/BIC (pas dans leur niveau absolu). Les conclusions gagnent à être croisées avec une **évaluation de type goodness-of-fit** : comparer, entre réseaux simulés et observé, des distributions de degrés, distances géodésiques, motifs triadiques, etc.