Smart-pooling et interactomes

Mémoire présenté pour l'obtention du

Diplôme d'Habilitation à Diriger des Recherches (DHDR)

Spécialité Biologie

Université de Grenoble

par

Nicolas Thierry-Mieg

Soutenue publiquement le 25 janvier 2013 à Grenoble
devant le jury composé de:

Philippe Cinquin, président
Yves Jacob, rapporteur
Eric Rivals, rapporteur
Jacques van Helden, rapporteur
Emmanuel Barillot, membre du jury
Eric Tannier, membre du jury

Sommaire

1. Introduction

Avant-propos

En tant que plus-si-jeune chercheur on espère avoir acquis une certaine maîtrise pour écrire des articles, présenter ses travaux ou encore former des étudiants, mais on peut encore se poser des questions sur ce qui est attendu dans le cadre de l'Habilitation à Diriger des Recherches. Heureusement les réponses se trouvent dans une circulaire du 16 novembre 1992, qui indique que l'habilitation à diriger des recherches « sanctionne la reconnaissance d'un haut niveau scientifique, du caractère original d'une démarche, de la maîtrise d'une stratégie de recherche dans un domaine large, de la capacité à encadrer de jeunes chercheurs. Le dossier de candidature est basé sur des ouvrages et travaux publiés accompagnés d'une synthèse permettant de faire apparaître l'expérience du candidat dans l'animation d'une recherche. L'habilitation n'est donc pas une thèse. Il s'agit d'une procédure qui doit, certes, être organisée de manière à garantir la haute qualité scientifique des candidats mais qui doit rester légère. On ne saurait, en particulier, exiger du candidat des conditions de délai pour leur inscription ou la préparation de l'habilitation dans l'établissement où celle-ci doit être présentée. On ne saurait non plus exiger du candidat la rédaction d'un véritable mémoire ni d'une seconde thèse, après celle du doctorat. »

Dans cet esprit, j'ai rédigé un manuscrit qui se veut synthétique et évite de trop rentrer dans les détails techniques – mes publications sont là pour ça, et sont disponibles sur ma page web, ainsi qu'en annexe de ce manuscrit pour les articles sur le thème « smart-pooling ». J'ai aussi fait le choix de présenter mes travaux dans leur diversité, plutôt que de me focaliser sur un sujet unique. Ainsi, s'il n'en faut qu'un le smart-pooling est clairement « mon » sujet principal depuis une dizaine d'années; il occupe donc une place prépondérante dans ce manuscrit, y compris dans son titre; mais je présente aussi les autres sujets sur lesquels j'ai travaillé, en précisant la nature de ma contribution sur chaque projet.

Démarche et thèmes de recherche

Après une formation initiale d'ingénieur en informatique, j'ai souhaité me tourner pour ma thèse vers la biologie, qui entrait dans l'ère génomique. J'ai alors rencontré Marc Vidal, qui démarrait au Massachussetts General Hospital de Boston un groupe dont l'objectif était d'identifier de manière systématique les interactions protéine-protéine par la méthode double hybride (yeast two-hybrid, Y2H), et qui m'a proposé de co-encadrer ma thèse. Ainsi, j'ai mené mes travaux de thèse conjointement entre son laboratoire à Boston, où j'ai séjourné plusieurs mois par an, et le laboratoire Logiciels-Systèmes-Réseaux (LSR) à Grenoble où mon

responsable de DEA Laurent Trilling soutenait ma reconversion à la bioinformatique avec enthousiasme. L'immersion dans l'équipe de Marc Vidal a confirmé mon goût pour la biologie. J'ai pu acquérir des connaissances spécifiques en biologie et mes compétences en informatique m'ont permis de contribuer de manière déterminante aux projets de clonage d'ORFéomes (Reboul et al, Nat Genet 2001) et de cartographie d'interactomes (Walhout et al, Science 2000; Davy et al, EMBO Rep 2001) du laboratoire. Mon travail de thèse a également comporté un volet plus théorique, concernant la prédiction d'interactions protéine-protéine (Thierry-Mieg et Trilling, Lecture Notes in Computer Science 2001). Les méthodes informatiques relevaient de la fouille de données (KDD, datamining) et de la Programmation Logique Inductive. D'un point de vue applicatif, l'objectif principal était d'accélérer la découverte d'interactions protéine-protéine en prioritarisant les expériences de double hybride. J'ai alors été recruté comme chargé de recherches au CNRS, dans une section de biologie (section 21: bases moléculaires et structurales des fonctions du vivant) mais affecté dans un laboratoire d'informatique (le LSR).

Ce parcours a conditionné ma démarche scientifique ultérieure: la biologie est la source des questions auxquelles je m'intéresse et le terrain d'application des solutions que je peux proposer, tandis que l'informatique et les mathématiques constituent pour moi des champs conceptuels et des outils qui me permettent de développer et de mettre en œuvre ces solutions. J'estime important de souligner que, bien que j'aie récemment effectué personnellement un travail de biologie “humide”, dans le cadre de la validation de l'approche smart-pooling pour la cartographie d'interactomes par Y2H, je suis fondamentalement un bioinformaticien, c'est-à-dire à mon sens un “biologiste sec”. Pour que ma recherche reste pertinente, il m'a toujours semblé indispensable d'établir des collaborations avec des biologistes expérimentaux.

Lors de mon recrutement au CNRS, j'avais donc développé une méthode de prédiction d'interactions protéine-protéine, dont la vocation était de prioritariser les expériences Y2H. Cependant, les progrès techniques très rapides ont rendu cette approche moins nécessaire: cribler un appât contre un ORFéome entier était devenu routinier dans les laboratoires tels que celui de Marc Vidal. Le problème principal qui semblait difficilement soluble à moyen terme était plutôt celui du bruit (faux positifs et faux négatifs) que celui du débit. J'ai donc réorienté mes recherches, sur ce qui est devenu l'approche “smart-pooling” (également appelée “Combinatorial Group Testing” en mathématiques discrètes, comme j'ai pu le découvrir ultérieurement) pour la biologie à haut débit. Le smart-pooling est une méthodologie expérimentale susceptible d'améliorer l'efficacité, la sensibilité et la spécificité dans les projets de criblage à haut débit. L'idée est de construire et tester des pools redondants d' “objets” (clones, protéines, drogues...), tels que chaque objet soit présent dans plusieurs pools et donc testé plusieurs fois. L'objectif

est de construire astucieusement les pools de manière à ce que les objets positifs puissent être identifiés directement au vu des valeurs des pools, et ceci malgré l'occurrence de pools faux-positifs et faux-négatifs. Mon travail sur ce thème à comporté un volet mathématique/combinatoire (conception des pools), un volet informatique/algorithmique (comment interpréter les résultats de smart-pooling), et un volet expérimental (évaluation de la méthode dans le cadre de l'identification d'interactions protéine-protéine par Y2H, d'abord sur un échantillon de l'interactome humain, puis à plus grande échelle chez *C. elegans)*. J'ai encadré entre 2002 et 2006 quatre étudiants de niveau M1 ou M2 sur ces sujets. La partie expérimentale a été réalisée en collaboration avec les laboratoires de Marc Vidal à Boston et Charlie Boone à Toronto. J'ai également encadré en 2011 une doctorante de University of California Riverside, USA, venue séjourner trois mois dans mon équipe à la recherche d'un nouveau projet de recherche pour orienter la fin de sa thèse. Elle s'intéresse aux liens entre smart-pooling et théorie de l'information de Shannon, et nous avons obtenu des résultats intéressants qu'elle poursuit actuellement.

J'ai également développé à partir de 2005 une solide collaboration avec Sylvie Ricard-Blum, de l'Institut de Biologie et Chimie des Protéines (IBCP) à Lyon, sur les interactions biomoléculaires dans la matrice extra-cellulaire. Je suis impliqué en particulier dans la conception et l'implémentation de la base de données MatrixDB (http://matrixdb.ibcp.fr), ainsi que dans les analyses bioinformatiques qui en découlent. J'ai ainsi co-encadré les travaux d'Emilie Chautard sur ce thème depuis 2005: stage d'ingénieur, puis M2 Recherches, et enfin sa thèse, soutenue en 2010. Au-delà du service rendu à la communauté scientifique par la constitution et la mise à disposition de cette ressource, qui nous a valu d'intégrer le consortium international IMEx qui fédère les principales bases de données d'interactions moléculaires, MatrixDB nous permet d'étudier de manière intégrative des questions biologiques spécifiques. Les problématiques biologiques auxquelles nous nous sommes intéressés comprennent l'initiation de l'angiogénèse, le vieillissement, et les interactions *Leishmania*-matrice (protéines et parasites entiers, plusieurs souches).

De manière plus large, je m'intéresse aux méthodes d'identification des interactions protéine-protéine et à l'analyse des réseaux d'interaction. Ainsi, j'ai encadré les travaux de M2 de Laure Sambourg en 2008-2009: nous avons développé une méthode originale pour évaluer la complétude des jeux de données d'interactions protéine-protéine actuellement disponibles, et estimer la taille de l'interactome de *S. cerevisiae*. Laure poursuit en thèse avec moi, et a d'abord approfondi puis publié ces résultats fin 2010. Elle s'est ensuite attaqué à un nouveau sujet, où elle étudie les liens entre interactome, épissage alternatif et cancer. Elle a effectué un séjour de six mois (juin à décembre 2011) dans l'équipe de Gary Bader à Toronto, avec qui nous collaborons dans le cadre de ce projet.

Plus généralement, je m'intéresse à l'analyse et à l'intégration de données post-génomiques, issues

notamment de la génomique fonctionnelle, de la transcriptomique, de la protéomique... J'ai ainsi collaboré de manière ponctuelle avec l'équipe de Goran Hansson en Suède, sur une étude des relations entre inflammation et insulino-résistance chez la souris. Ma contribution a consisté à analyser et comparer plusieurs jeux de données d'expression de gènes. J'ai aussi co-encadré avec Marc Vidal (Boston) et Laurent Trilling (Grenoble) la thèse d'Anne-Ruxandra Carvunis, soutenue début 2011. Anne-Ruxandra a contribué de manière essentielle aux grands projets interactome du laboratoire de Marc Vidal, ce qui lui a valu plusieurs articles en tant que (co-)premier auteur au cours de sa thèse (Genome Res 2008, Nat Methods 2009, et deux Science 2011), mais j'ai refusé de co-signer ces articles car j'ai estimé que ma contribution à ces travaux n'était pas suffisante. En revanche j'ai contribué de manière plus importante à son projet propre, qui a consisté à étudier les gènes peu conservés chez la levure *S. cerevisiae* et les mécanismes de naissance *de novo* des gènes. Ces travaux ont conduit à une co-publication cette année (Carvunis et al, Nature 2012).

Les technologies de séquençage seconde génération (NGS) vont transformer des pans entiers de la recherche biomédicale – elles ont par exemple déjà révolutionné l'étude du transcriptome et de son expression. Il est donc important d'apprendre à analyser et traiter correctement les données NGS, et de développer des algorithmes spécifiques pour ces analyses si les solutions actuellement existantes ne sont pas satisfaisantes. Je me suis engagé récemment dans cette voie, d'abord à travers les travaux de thèse de Laure Sambourg dont le projet repose sur l'analyse de données NGS de génomes, transcriptomes et exomes de cancers, mais également dans le cadre d'une collaboration naissante avec François Parcy (iRTSV/PCV, Grenoble) où nous analysons des données ChIP-seq pour l'étude des interactions protéine-ADN chez *Arabidopsis thaliana*, et en collaboration avec Pierre Ray (Institut de Biologie et de Pathologie du CHU et laboratoire AGIM, Grenoble) pour l'analyse de données exome-seq de patients souffrant de diverses formes d'infertilité masculine. Cette dernière collaboration a récemment donné lieu à une co-publication soumise, dont je suis avant-dernier auteur.

Collaborations principales

• Marc Vidal, Center for Cancer Systems Biology, Dana Farber Cancer Institute, Boston, USA. Publications communes, co-encadrement d'une doctorante.

• Sylvie Ricard-Blum, Institut de Biologie et Chimie des Protéines, Lyon, France. Publications communes, co-encadrement d'une doctorante.

• Charlie Boone, Terrence Donnelly Centre for Cellular and Biomolecular Research, University of Toronto, Canada. Publications communes.

• Goran Hansson, Center for Molecular Medicine and Department of Medicine, Karolinska University Hospital Solna, Sweden. Publication commune.

• Gary Bader, Terrence Donnelly Centre for Cellular and Biomolecular Research, University of Toronto, Canada. Collaboration en cours, projet "Cancer Interactomes".

• Pierre Ray, Institut de Biologie et de Pathologie du CHU, et laboratoire AGIM, Grenoble, France. Publication commune soumise, collaboration en cours (exome-seq infertilité masculine).

• François Parcy, iRTSV/PCV, Grenoble, France. Collaboration en cours (chip-seq *Arabidopsis thaliana*).

2. Smart-pooling pour le criblage haut-débit

Contexte

La connaissance de génomes, transcriptomes ou protéomes complets a ouvert de nouvelles possibilités, permettant en principe “d'interroger” tous les gènes ou leurs produits d'un seul coup. Notamment, certains projets à grande échelle cherchent à identifier les rares molécules qui sont positives dans un test binaire. C'est le cas par exemple des projets d'identification systématique d'interactions protéine-protéine par double hybride. L'efficacité des protocoles (coût) est évidemment une préoccupation essentielle, mais le bruit inhérent aux expériences à haut-débit est également un souci majeur: il génère à la fois des faux positifs et des faux négatifs, et la reproductibilité des expériences n'est pas acquise a priori.

En pratique, un protocole largement employé repose sur une stratégie de pooling naïve à deux étapes: on effectue initialement les tests sur des pools de molécules, puis on reteste individuellement toutes les molécules présentes dans les pools positifs. Comparé au test individuel de chaque molécule, ce protocole améliore l'efficacité, mais il n'apporte rien pour la détection des faux-négatifs, qui nécessitent la répétition de l'expérience complète pour être identifiés. J'ai donc cherché une autre solution, dans le but d'améliorer à la fois l’efficacité, la robustesse et la sensibilité des projets de criblage à haut débit.

Une stratégie prometteuse pour atteindre ces objectifs est la méthode du “smart-pooling” (Thierry-Mieg, Nat Methods 2006). Il s'agit de tester des pools redondants de molécules construits astucieusement, de sorte qu'en observant les résultats des tests sur les pools, on puisse directement identifier les positifs et corriger les erreurs (faux positifs et faux négatifs). Un système de smart-pooling rudimentaire bien connu consiste à répartir conceptuellement les molécules dans un tableau, puis à tester les lignes et les colonnes, comme illustré **Figure 1**. Des variantes de cette construction sont couramment utilisées en biologie. Cependant, ces stratégies sont typiquement très vulnérables au bruit expérimental et se comportent mal lorsque plus d'un objet est positif, et ceci bien qu'elles proposent un nombre relativement important de tests. Or, la théorie de l'Information de Shannon suggère que de très bonnes constructions peuvent exister: le système “lignes-et-colonnes” illustre bien le concept, mais on peut faire beaucoup mieux.

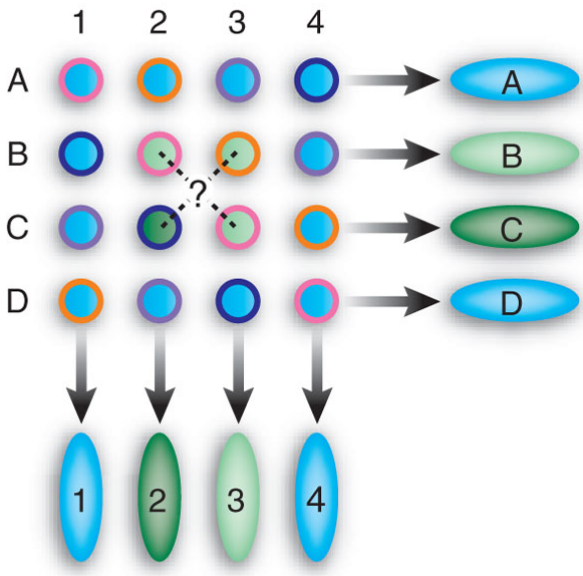

Figure 1: Exemple de système de "smart-pooling". Seize "proies" sont réparties dans un quadrillage imaginaire (positions A1 à D4), et combinés pour produire 8 pools (un par ligne: A-D, et un par colonne: 1-4). Chaque pool contient 4 proies. Si les pools sont testés contre un "appât" en l'absence de bruit, et que seuls les pools C et 2 sont positifs (vert), alors C2 est l'unique proie positive. Mais si les pools B et 3 sont également positifs (vert clair), les deux solutions possibles (B2 et C3 positifs) ou (C2 et B3 positifs) ne peuvent être distinguées. Ceci peut être résolu en ajoutant quatre pools, construits selon les diagonales du quadrillage comme indiqué par les couleurs qui entourent les proies. La continuité des diagonales peut être visualisée en enroulant la figure autour d'un cylindre. Si la diagonale rose est positive, (B2 et C3) est la solution, tandis que si les pools diagonaux orange et bleu sont positifs, la solution est (C2 et B3).

Pooling problem et solution mathématique: STD

Ce problème peut être formalisé en termes mathématiques, et une petite communauté de combinatoriciens et informaticiens s'y intéresse. Toutefois, ils sont assez éloignés de la réalité expérimentale, et les solutions qu'ils proposent, si elles ont typiquement de bons comportements asymptotiques, sont rarement efficaces en pratique. Ainsi, certains de ces systèmes sont très performants, c'est-à-dire qu'ils nécessitent peu de tests, mais ils manquent de souplesse et de robustesse pour des applications concrètes en biologie. D'autres possèdent ces deux qualités, mais leurs performances laissent à désirer. Mon premier objectif a été de concevoir et développer un système de pooling correcteur d'erreurs qui combine flexibilité, robustesse et performance.

Mes recherches ont abouti à une méthode originale de smart-pooling: le "Shifted Transversal Design" (STD). La construction est extrêmement souple: en jouant sur les valeurs de ses trois paramètres, elle peut être appliquée pour identifier un nombre quelconque de positifs et pour corriger un taux élevé d'erreurs si nécessaire. Néanmoins, elle se révèle très performante en termes de nombre de tests: son efficacité est typiquement bien supérieure à celles des systèmes précédemment décrits dans la littérature.

Les propriétés principales de STD (identification des positifs, correction d'erreurs, performances) ont été démontrées formellement, en faisant appel à l'algèbre de Galois et à l'arithmétique. Après plusieurs présentations dans des conférences à partir de 2003, ces travaux ont été enfin publiés en 2006 (Thierry-Mieg, BMC Bioinformatics 2006). Plutôt que de rentrer ici dans les détails, je renvoie le lecteur mathophile à l'article (disponible sur ma page web, comme la plupart de mes publications, et aussi en

annexe A2 de ce document), où les aspects mathématiques sont exposés de manière claire (et élégante, à mon subjectif avis!). La construction STD est illustrée de manière simplifiée **Figure 2**, et à plus grande échelle **Figure 3**.

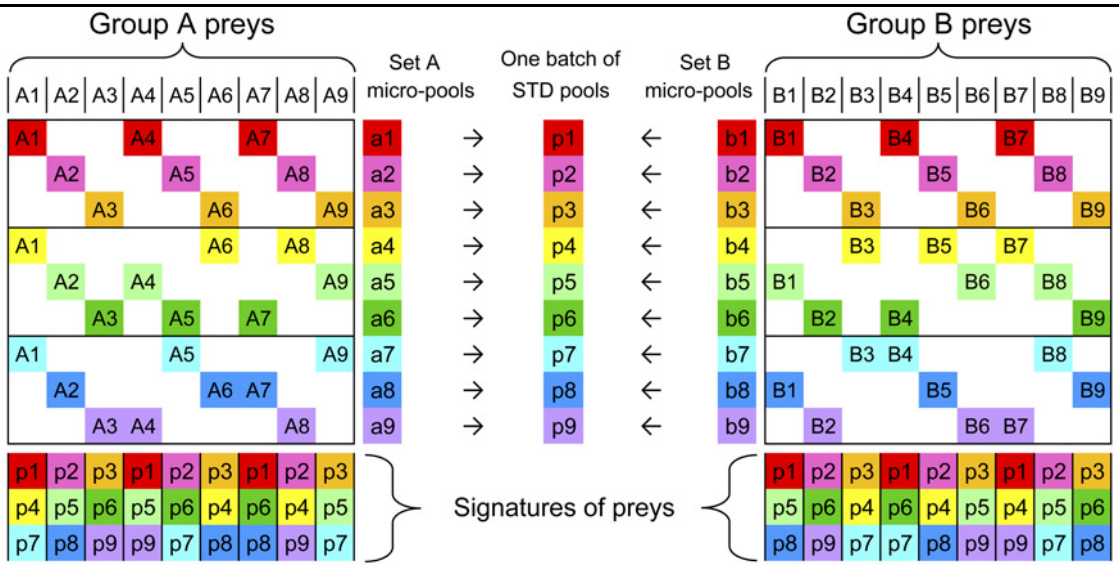

Figure 2: Une construction STD simple. Deux groupes de neuf proies (groupe A: A1-A9 et groupe B: B1-B9) sont poolés séparément dans neuf micropools (set A: a1-a9 et set B: b1-b9), selon des sous-systèmes d'un système STD plus grand (capable de prendre en compte les 18 proies). Chaque micro-pool contient trois proies (par exemple a1 contient A1, A4 et A7); et chaque proie est présente dans une combinaison unique de trois micro-pools (par exemple A4: a1/a5/a9). Les paires de micro-pools des sets A et B de même indice peuvent être superposés pour générer des pools STD (un “batch”: p1-p9), contenant chacun 6 proies, et chaque proie possède une signature unique dans les pools STD.

Smart-pooling en pratique: interprétation des résultats, le logiciel Interpool

La formalisation mathématique du “pooling problem” comprend des hypothèses et des conclusions très fortes. Ainsi, on suppose que les nombres de positifs et d'erreurs ne dépassent en aucun cas des seuils fixés. En échange, si cette hypothèse est satisfaite, on garantit que toutes les erreurs et tous les positifs peuvent être identifiés. Ce cadre exigeant est très utile, dans la mesure où il permet de raisonner de manière rigoureuse et de comparer objectivement les systèmes de pooling. Cependant, pour des applications concrètes il est souhaitable d'alléger ces contraintes. D'une part, il faudrait choisir des seuils très élevés pour être sûr qu'ils ne sont jamais franchis. D'autre part, on pourrait tolérer que quelques positifs ne soient exceptionnellement pas identifiés, si cela permet de construire et cribler beaucoup moins de pools. Par exemple, on pourrait être confronté au choix suivant: soit tester 1000 pools en ayant une garantie mathématique que, si le comportement du système biologique est conforme à la spécification du modèle, tous les objets positifs seront identifiés; soit tester 500 pools, en sachant qu'en moyenne 99% des positifs seront reconnus. La seconde solution sera généralement préférée.

Notons que cette considération ne remet pas en cause le choix du système de pooling: une construction efficace dans le cadre formel sera également performante dans le cadre allégé. En revanche, l'interprétation des résultats devient problématique: l'algorithme classique repose sur la condition de garantie et sur la connaissance des seuils; il n'est donc plus exploitable. Le seul algorithme publié jusqu'en 2008 qui n'ait pas ces limitations repose sur une méthode de Monte Carlo sur chaînes de Markov (logiciel MCPD). Cet algorithme ne nous convenait pas, car sa nature ne lui permet de fournir aucune garantie sur la qualité de ses résultats, et ses performances sont trop limitées pour réaliser des campagnes de simulation avec une couverture satisfaisante. J'ai donc cherché à développer une autre solution: j'ai conçu et implémenté un algorithme original, dont les résultats sont garantis et dont les performances s'avèrent excellentes. L'algorithme, de type "branch and bound", est détaillé dans (Thierry-Mieg et Bailly, Bioinformatics 2008). Le logiciel, nommé Interpool (Interpretation of Pooling Results) et comportant environ 18000 lignes de code C, est distribué sous licence libre GNU GPL sur mon site web.

Interpool permet d'effectuer un grand nombre de simulations rapidement, et donc de choisir les paramètres de STD les mieux adaptés à un problème biologique donné, en fonction du nombre de variables, des taux de positifs et d'erreurs attendus, et du nombre de variables ambiguës que l'expérimentateur est prêt a tolérer. L'étape suivante consistait donc à valider l'ensemble de l'approche dans une expérience réelle.

Expérience pilote

Je collabore de manière plus ou moins suivie depuis ma thèse avec le laboratoire de Marc Vidal, qui dirige le Center for Cancer Systems Biology (CCSB) au Dana Farber Cancer Institute (DFCI), Boston. Un des principaux problèmes auxquels il est confronté pour la cartographie des interactomes est celui du bruit expérimental: comme toute méthode, le système double hybride produit des faux négatifs et des faux positifs; et en passant à l'échelle d'ORFéomes complets, ce problème devient critique. Estimant à juste titre que les faux positifs sont bien plus nuisibles, le laboratoire utilise des protocoles spécifiques qui cherchent à limiter ceux-ci au maximum; mais du coup, les faux négatifs sont très fréquents: les cartes d'interaction obtenues sont loin d'être complètes. La solution triviale consisterait à répéter chaque expérience plusieurs fois, mais cette stratégie est très coûteuse. Le laboratoire bénéficierait d'une méthode qui permette d'identifier et de corriger les deux types d'erreurs tout en minimisant le travail expérimental nécessaire. C'est cette problématique qui m'a conduit à développer l'approche smart-pooling depuis 2002.

C’est donc assez naturellement que la première évaluation expérimentale de STD a été conduite au DFCI/CCSB. En 2005, j’ai utilisé leurs robots pour construire 169 pools à partir de 940 clones de l’ORFéome humain, selon un système STD similaire à celui présenté **Figure 3**: chaque clone est présent dans 13 pools différents, ce qui permet de corriger de nombreuses erreurs. En collaboration avec Jean-François Rual, nous avons criblé 100 appâts contre les 169 pools, ce qui représente 94000 paires de protéines explorées. Ces paires avaient été précédemment testées avec le protocole standard du CCSB. Les résultats sont convaincants: plus de 80% des interactions connues ont été retrouvées, et de nombreuses nouvelles interactions, fausses négatives dans leur protocole haut débit standard, ont été détectées. Au final, la méthode smart-pooling présente une sensibilité trois fois supérieure au protocole standard. Ces résultats constituent une partie de l'article sur l'évaluation de l'approche STD (Xin et al, Genome Research 2009).

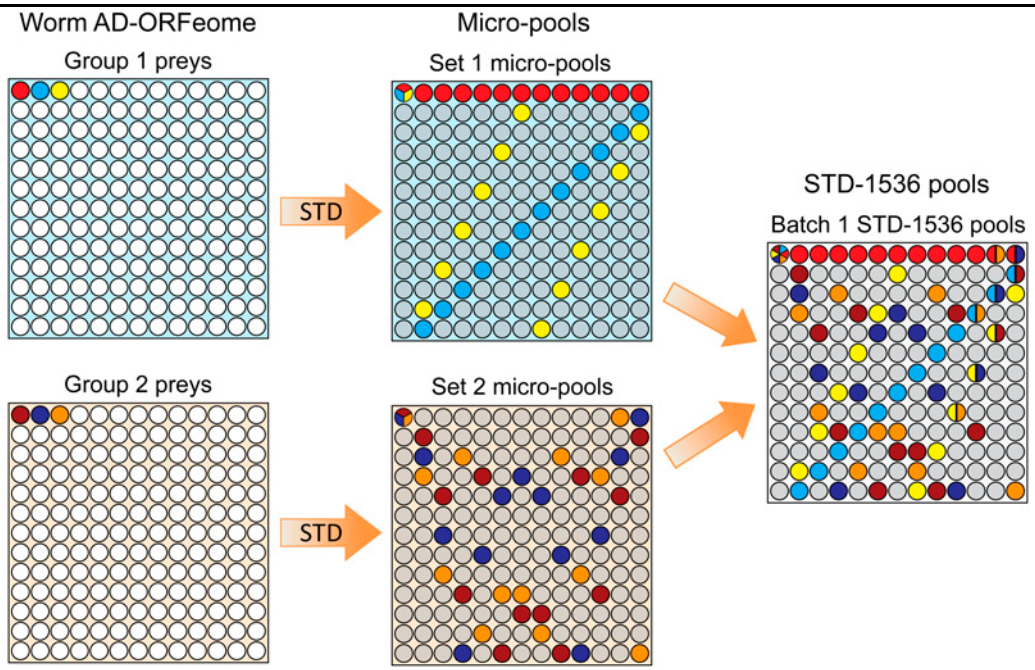

Figure 3: Construction STD utilisée pour l'évaluation à l'echelle d'un ORFéome. 12.675 proies de l'ORFéome de *C. elegans* ont été répartis en 75 groupes contenant chacun 169 proies (deux groupes sont représentés ici). Chaque group a été poolé selon STD en 169 micro-pools. Chaque micro-pool contient 13 proies, et chaque proie est contenue dans une unique combinaison de 13 micro-pools, comme illustré par trois proies dans chaque groupe (couleurs). Deux proies sont présentes de façon concomittante dans au plus un micro-pool, de telle sorte que chaque proie est définie de manière unique par n'importe quels deux des treize micro-pools qui la contiennent. De plus, les signatures des proies de même coordonnées dans différents groupes de micro-pools sont très différentes (par exemple rouge clair et rouge foncé dans les groupes 1 et 2). Ainsi les sets de micro-pools peuvent être superposés pour obtenir des smarts-pools efficaces. Par exemple, si on superpose jusqu'à 13 groupes de micro-pools (deux sont superposés à droite), toute proie reste définie de manière unique par n'importe quels trois des treize smart-pools qui la contiennent.

Evaluation à l'échelle d'un ORFéome complet

Encouragés par les résultats obtenus avec STD dans l'expérience pilote, nous avons réalisé une évaluation à plus grande échelle, en collaboration avec Xiaofeng Xin et Charlie Boone de l'Université de Toronto. Ils disposent dans leur laboratoire de robots extrêmement miniaturisés: alors que Marc Vidal travaille au format 96 et un peu 384, Charlie Boone utilise les formats 768, 1536, et jusqu'à 6144. Nous avons décidé d'appliquer l'approche STD à un organisme complet (*C. elegans*), tout en évaluant la capacité de la méthode à tirer parti de ces formats haute-densité. La première étape expérimentale a été réalisée à Boston en août-septembre 2006: Xiaofeng et moi-même avons construit des smart-pools à partir des 12675 ORFs clonés de *C. elegans*. Un point important dans la construction des pools est qu'on souhaitait pouvoir utiliser les mêmes pools pour plusieurs formats, et donc avec des tailles de pools différentes (plus le format est dense, plus les pools doivent être petits). Grâce à la symétrie de STD, cet objectif est en fait assez facile à atteindre: on construit des “micro-pools”, qui sont ensuite simplement superposés de diverses manières afin d'obtenir des smart-pools aux tailles voulues, comme illustré **Figures 2 et 3**.

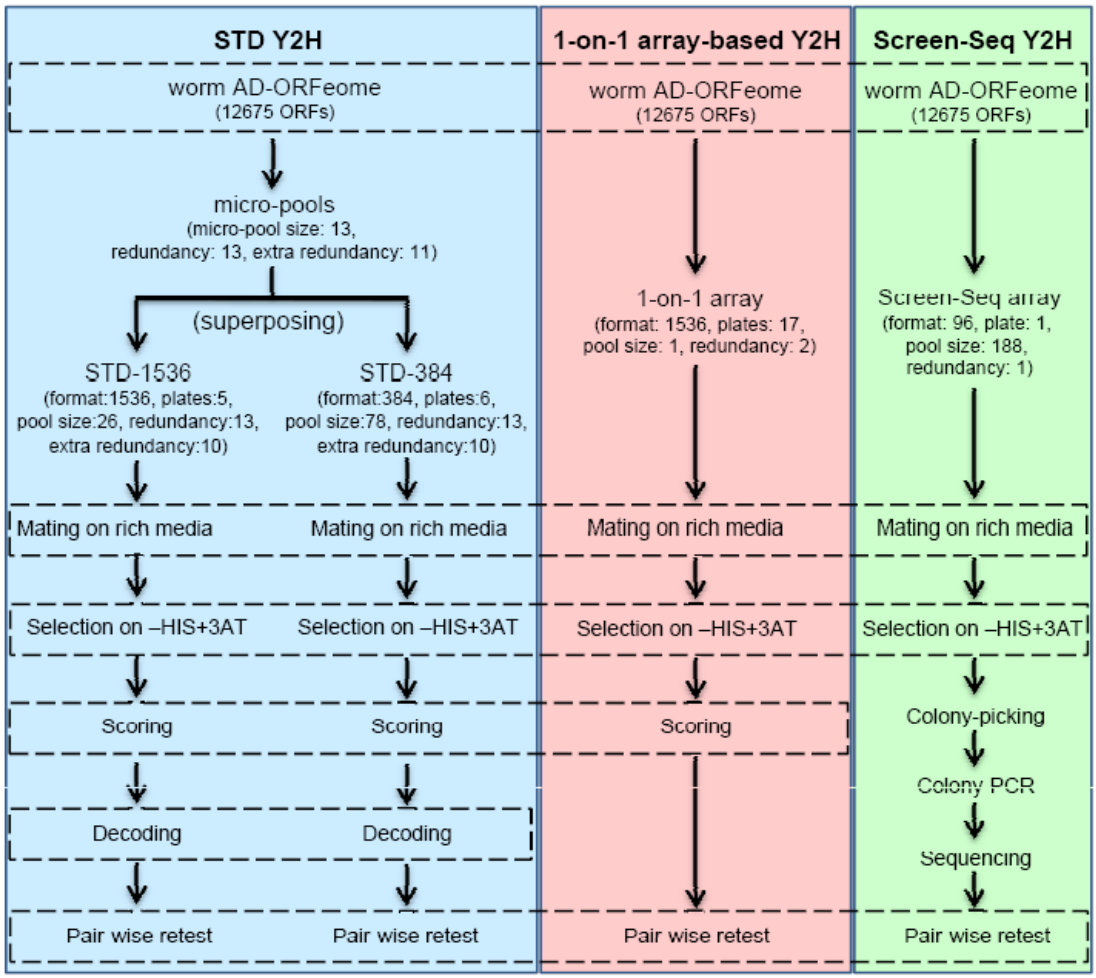

Figure 4: Vue schématique de la validation expérimentale sur l'ORFéome de *C. elegans*. La figure compare les étapes des méthodes Y2H utilisées: smart-pooling aux formats 1536 et 384 (STD), tests individuels dupliqués (1-on-1), et criblage de mini-pools suivi d'identification des positifs par séquençage (Screen-Seq).

Notons que cette « modularité » de STD constitue une piste de recherches intéressante sur le plan théorique, que nous poursuivons en collaboration avec Atri Rudra (U. of Buffalo) et Anna Gilbert (U. of Michigan).

Il s'agissait ensuite d'utiliser ces smart-pools dans des cribles double-hybride. Nous avons criblé douze appâts contre les smart-pools, à la fois au format 384 (avec des pools de taille 78) et au format 1536 (avec des pools de taille 26). Nous avons également criblé ces mêmes appâts contre l'ORFéome en individuel (un contre un en format 1536, spots dupliqués), méthode lourde et coûteuse mais a priori très sensible; ainsi qu'avec la méthode classique du CCSB (mini-pools de 188 proies en format 96, identification des positifs par séquençage). Les différentes étapes des trois méthodes sont représentées **Figure 4**. De plus tous les cribles ont été dupliqués, afin de pouvoir évaluer la robustesse des méthodes. Ces travaux ont confirmé les qualités de l'approche STD, qui atteint des niveaux de sensibilité et de spécificité similaires à la méthode un contre un, pour un coût (financier et humain) trois fois moindre. La méthode classique du CCSB (Screen-Seq) reste trois fois moins coûteuse que STD, mais elle est également deux fois moins sensible. Screen-Seq reste donc une approche raisonnable pour obtenir rapidement des interactomes de faible couverture, tandis que STD ressort comme la meilleure méthode pour produire des cartes plus denses. Notons que si la méthode Screen-Seq est répliquée trois fois pour gagner en sensibilité, comme l'a fait le CCSB pour son récent interactome de *S. cerevisiae* (Yu et al, Science 2008), la sensibilité n'augmente que de 30% alors que le coût devient similaire à celui de STD (qui offre 100% d'augmentation). Ces résultats ont été publiés dans un article dont je suis dernier mais aussi co-premier auteur (Xin et al, Genome Research 2009). Je voudrais également signaler que l'intégralité des données brutes et des logiciels d'analyse utilisés a été packagée et soigneusement documentée, afin de permettre la reproduction des analyses et la compréhension de notre travail dans les moindres détails. Le package constitue un Supplementary Data de l'article, et est également téléchargeable sur ma page web. Un second article récent décrit plus en détail les protocoles (Xin, Boone, Thierry-Mieg; Methods in Molecular Biology 2012).

Conclusions et perspectives

Je tiens à souligner que mes travaux sur ce sujet sont vraiment pluridisciplinaires: la définition et l'étude formelle du système STD repose sur l'arithmétique; le développement des algorithmes d'Interpool est typiquement un travail d'informatique; et la conception et la mise en œuvre des expériences pilote et ORFéome-scale constituent mes premiers travaux personnels de biologie humide.

La publication de mes résultats, sur cette thématique originale et méconnue des biologistes comme des

informaticiens, a été initialement difficile. Ainsi l'article sur STD publié en janvier 2006 dans BMC Bioinformatics était essentiellement identique en 2004. D'un autre coté, les risques ont payé puisque mes premières publications sur le sujet m'ont placé au premier plan sur la scène internationale dans le domaine. Ainsi, j'ai été invité à présenter STD lors du “Combinatorial Group Testing Workshop” organisé au DIMACS Center (Discrete Mathematics and Theoretical Computer Science, New Jersey) en mai 2006. Ce workshop rassemblait les mathématiciens et informaticiens intéressés par les aspects théoriques du “pooling problem”, et ceux-ci ont confirmé les qualités du système STD du point de vue formel. J'ai également été sollicité par Nature Methods pour introduire et discuter l'approche dans un News&Views, qui a officialisé et généralisé l'appellation “smart-pooling”. Par la suite, j'ai été invité à présenter l'approche assez largement: Grenoble, Marseille, Hinxton, Toronto, Minneapolis, Tsukuba, Nagoya. J'ai également co-organisé avec A. Schliep (à l'époque au Max Planck à Berlin) et A. Shokrollahi (EPFL Lausanne) le Dagstuhl Seminar “Group Testing in the Life Sciences”, à Dagstuhl (Allemagne) en juillet 2008. Les Dagstuhl Seminars sont des petites conférences internationales très prestigieuses en informatique, similaires aux Banbury Meetings de CSHL en biologie moléculaire et génétique.

Au niveau expérimental, l'approche smart-pooling avec STD me semble mûre, et peut être mise à profit dans de nombreuses applications. En effet, bien que ce travail ait été initialement motivé par le projet de cartographie des interactions entre protéines du CCSB, l'approche est applicable bien au-delà du double-hybride à haut débit, comme par exemple dans les projets reposant sur des tests PCR, ou encore dans le domaine du criblage de chimiothèques (voir la revue de Kainkaryam et al, Curr Opin in Drug Disc 2009). Les technologies de séquençage de seconde génération (454, Solexa, SoLiD) ouvrent aussi de nouvelles possibilités d'application, en combinant smart-pooling et barcoding comme l'ont montré deux publications en 2009 (Erlich *et al* et Prabhu *et al*, Genome Research, July 2009). Lors d'un workshop à Minneapolis en février cette année, j'ai appris avec plaisir que STD était actuellement utilisé dans un projet de séquençage *de novo* de l'orge (Stefano Lonardi, UC Riverside), ainsi que pour la recherche d'allèles rares par séquençage (Yaniv Erlich, Whitehead Institute, qui a abandonné son système de pooling décrit dans l'article de 2009 sus-cité pour adopter STD, plus puissant et plus souple). STD a également été utilisé par exemple pour le criblage de banques de BACs (Wu et al. J Bioinform Comput Biol 2008, 6(3):603-22), le profilage de délétants de *S. pombe* (Han et al. Genome Biology 2010, 11:R60) et pour la recherche de drogues synergistiques (Severyn et al. ACS Chemical Biology 2011, 6(12):1391-8), et j'ai bon espoir que la méthode soit employée dans d'autres applications dans les années qui viennent.

Sur le plan théorique en revanche, plusieurs aspects méritent d'être approfondis. Tout d'abord, la modularité de STD s'est révélée très utile dans nos expériences ORFeome-wide (Xin et al, Genome Research 2009). Suite au workshop de Minneapolis où j'ai insisté sur cette modularité, nous avons

commencé à formaliser et analyser le problème mathématique sous-jacent, en collaboration principalement avec Atri Rudra (U. of Buffalo) et Anna Gilbert (U. of Michigan). Par ailleurs, je soupçonnais depuis des années l'existence d'une connexion forte entre STD et les célèbres codes correcteurs d'erreurs de Reed-Solomon, sans arriver à l'établir. Au cours de ce même workshop à Minneapolis, nous sommes parvenus à expliciter cette connexion, en collaboration principalement avec encore Atri Rudra (U. of Buffalo) ainsi que Or Zuk (Broad Institute), et un article est en préparation. Enfin, les liens entre smart-pooling et théorie de l'information de Shannon pourraient être approfondis. C'est encore une problématique qui m'intéresse depuis plusieurs années mais sur laquelle je n'avais pas pu me focaliser par manque de temps. Denise Duma, doctorante à UC Riverside (directeur: S. Lonardi), est venue travailler avec moi sur ce thème pendant trois mois en 2011, et j'espère que cette collaboration pourra aboutir à des découvertes intéressantes.

3. Interactome de la matrice extra-cellulaire

Je participe depuis 2005 à un projet avec Sylvie Ricard-Blum, de l'IBCP de Lyon: il s'agit d'identifier et d'étudier les interactions entre biomolécules (protéines, fragments protéiques, sucres, lipides) qui ont lieu dans la matrice extra-cellulaire, en nous concentrant initialement sur plusieurs domaines de collagènes humains.

Les collagènes forment une superfamille de protéines qui sont les plus abondantes de l'organisme (30% des protéines totales). Ils sont localisés à l'extérieur ou à la surface des cellules et sont impliqués dans l'assemblage et la fonction des matrices extra-cellulaires. Ils participent également aux interactions cellules-matrice qui modulent le comportement cellulaire. Certains collagènes contiennent des domaines ou des fragments libérés par protéolyse qui possèdent des activités biologiques qui leur sont propres. Ces domaines bioactifs contrôlent par exemple la biosynthèse et l'assemblage de protéines extra-cellulaires, l'adhésion, la prolifération ou la migration cellulaire et régulent des processus physio-pathologiques tels que l'angiogenèse, la croissance tumorale et la réparation tissulaire. Nous nous intéressons en priorité a certains domaines bioactifs qui régulent l'angiogenèse et la croissance tumorale et constituent des agents thérapeutiques potentiels, mais dont les mécanismes d'action sont mal connus. Comme la plupart des protéines, ces domaines exercent leurs rôles biologiques en interagissant physiquement avec d'autres biomolécules. Certains de leurs partenaires moléculaires et cellulaires ont été identifiés mais les seules interactions caractérisées pour l'instant ont été décrites entre deux partenaires seulement, c'est-à-dire entre un domaine ou un fragment de collagène et un ligand. Ces interactions binaires ne permettent pas de déterminer le mécanisme d'action de ces domaines dans un système intégré et complexe correspondant à la situation existant in vivo. L'objectif de notre projet est d'identifier et de caractériser les interactions de ces domaines dans leur contexte global, au sein du réseau complexe d'interactions protéine-protéine et protéine-glycosaminoglycane de la matrice extra-cellulaire pour déterminer leurs mécanismes d'action.

La première étape du projet consistait à répertorier et représenter l'ensemble des interactions établies par quatre domaines bioactifs des collagènes. A cette fin nous avons construit la base de données MatrixDB (http://matrixdb.ibcp.fr), qui va en fait bien au-delà de ces quatre domaines: elle inclut toutes les protéines de la matrice extra-cellulaire (dont les collagènes et leurs domaines bioactifs), les glycosaminoglycanes et leurs partenaires respectifs. Notre objectif est que MatrixDB soit la plus complète possible en terme d'interactions entre biomolécules dans la matrice extra-cellulaire. Les interactions proviennent de bases de données publiques, mais également de curation de littérature que nous réalisons ainsi que des résultats expérimentaux que nous obtenons à l'aide de la plateforme technologique basée sur la résonance plasmonique de surface (SPR: BiaCore, FlexChip) dont Sylvie Ricard-Blum est responsable. La topologie

du réseau global d'interactions des domaines bioactifs est ensuite analysée, par exemple pour identifier les ligands vers lesquels convergent le plus grand nombre d'interactions (hubs), identifier les sous-réseaux fortement connectés (clusters), ou pour déterminer si les domaines ou fragments régulant l'angiogenèse possèdent des ligands communs constituant une “signature d'interactions”. A ce stade nous retrouvons ou confirmons des associations connues sans faire de réelle découverte surprenante, mais nous continuons d'enrichir régulièrement le contenu de MatrixDB ce qui améliore la pertinence des résultats de nos analyses.

Nous projetons également d'exploiter le réseau global des interactions pour établir un modèle dynamique des évènements extra-cellulaires liés à l’angiogenèse, et plus particulièrement à l'angiogenèse tumorale. Nous avons collaboré sur cette thématique de modélisation avec Eric Fanchon, chercheur dans la même équipe que moi (BCM, laboratoire TIMC-IMAG). A ce stade cet aspect reste préliminaire, car les données disponibles semblent insuffisantes pour construire un modèle détaillé, mais nous poursuivons nos efforts et pourrions aboutir à moyen terme.

Mon rôle dans le projet consiste notamment à superviser les aspects bioinformatiques. J'ai co-dirigé avec Sylvie les travaux d'Emilie Chautard au cours de son projet d'ingénieur (2005-2006), puis de son Master 2

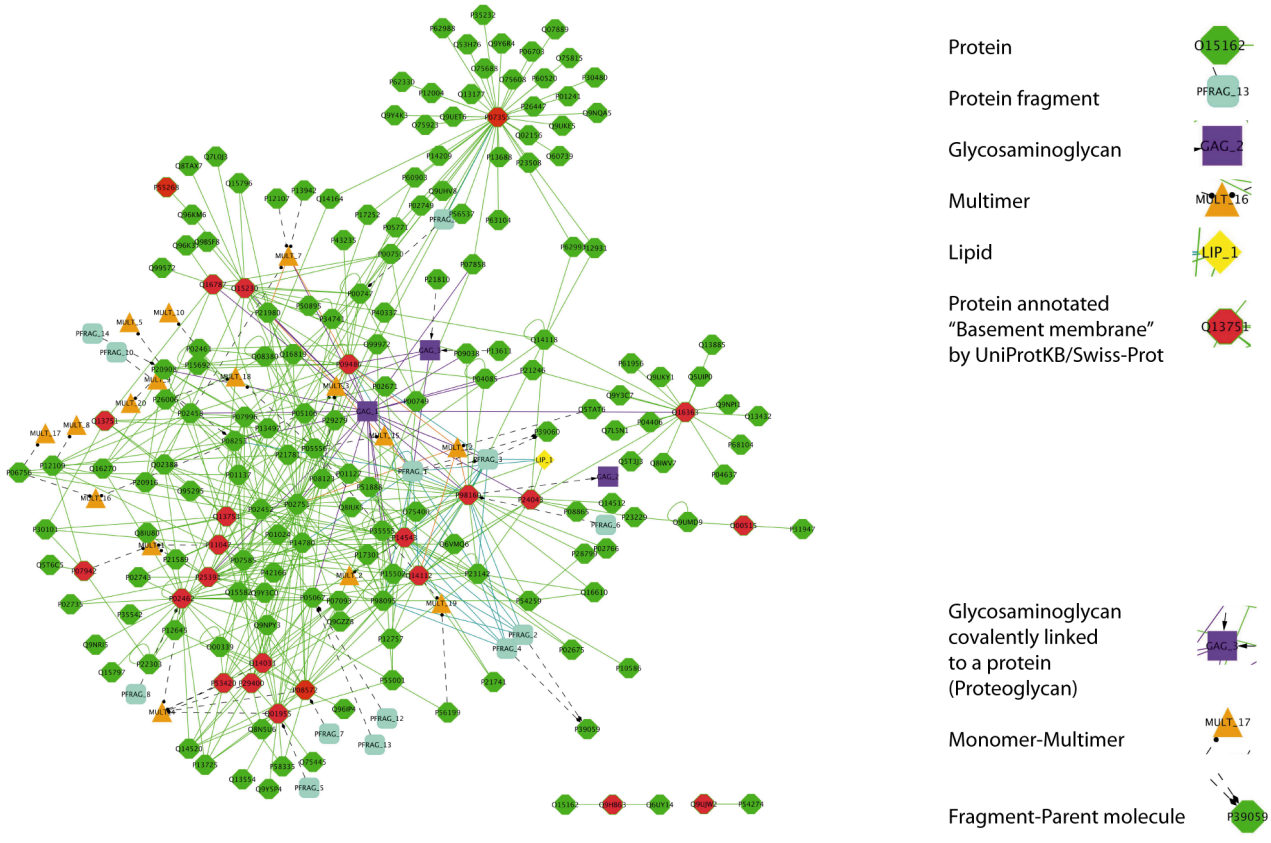

Figure 5: Réseau obtenu en effectuant une requête dans MatrixDB pour le mot-clé UniProtKB / Swiss-Prot "Basement membrane", visualisé avec Cytoscape.

Recherche (2006-2007), et enfin de sa thèse (2007-2010). Après une période de transition un peu difficile, Guillaume Launay a été recruté sur un poste de MCF dans l'équipe de Sylvie en 2011, et je travaille désormais avec lui sur l'amélioration et l'évolution de MatrixDB.

Ces travaux ont donné lieu à plusieurs présentations lors de conférences ainsi qu'à quatre articles (Chautard et al: Bioinformatics 2009, Pathol Biol 2009, Biogerontology 2010, Nucleic Acids Res 2011). MatrixDB est membre du consortium IMEx (International Molecular Exchange consortium), qui fédère les principales bases de données d'interactions entre biomolécules. En plus d'être explorables via l'interface web, notamment via une applet Cytoscape comme illustré **Figure 5** pour les protéines de la membrane basale, les données de MatrixDB sont distribuées librement dans les formats standards actuels (PSI-MI XML et TAB 2.5).

De plus nous avons récemment commencé à travailler sur un sujet connexe, en collaboration avec Philippe Esterre (institut Pasteur de Cayenne, Guyanne Française). La leishmaniose est considérée comme l'une des six maladies tropicales majeures dans les pays en voie de développement, selon l'Organisation Mondiale de la Santé. Notre projet consiste à identifier *in vitro* et *in silico* les interactions établies par les *Leishmania*, protozoaires responsables de la leishmaniose, avec la matrice extracellulaire de l'hôte. Cela permettra d'étudier les mécanismes à l'origine du tropisme tissulaire cutané ou viscéral des *Leishmania* et de modéliser les premières étapes de l'infection dans lesquelles ces interactions sont impliquées. Nous nous intéressons en particulier au rôle des protéine-kinases de *Leishmania* dans ces processus. Un petit financement (région Rhone-Alpes) a été obtenu sur ce sujet cette année.

Un autre aspect que nous voulons développer est l'intégration dans MatrixDB de données sur l'épissage alternatif et sur la tissue-spécificité de l'expression. Nous analyserons alors les interactomes extra-cellulaires à la lumière de ces nouvelles données. On pourra par exemple étudier l'interactome en se limitant aux molécules exprimées dans un tissu choisi.

4. Analyse de l'interactome de *S. cerevisiae*

J'encadre depuis 2009 la thèse de Laure Sambourg. En début de thèse, Laure a approfondi ses recherches sur l'interactome de la levure *S. cerevisiae*, débutées au cours de son stage de M2 que j'avais encadré l'année précédente, pour aboutir à une publication fin 2010 (Sambourg et Thierry-Mieg, BMC Bioinformatics 2010). Nous avons développé une méthode originale pour évaluer la complétude des données d'interaction disponibles et estimer la taille totale d'un interactome, en prenant la levure *S. cerevisiae* comme cas d'étude.

Nous avons examiné l'interactome de la levure sous un nouveau jour, en prenant en compte l'attention

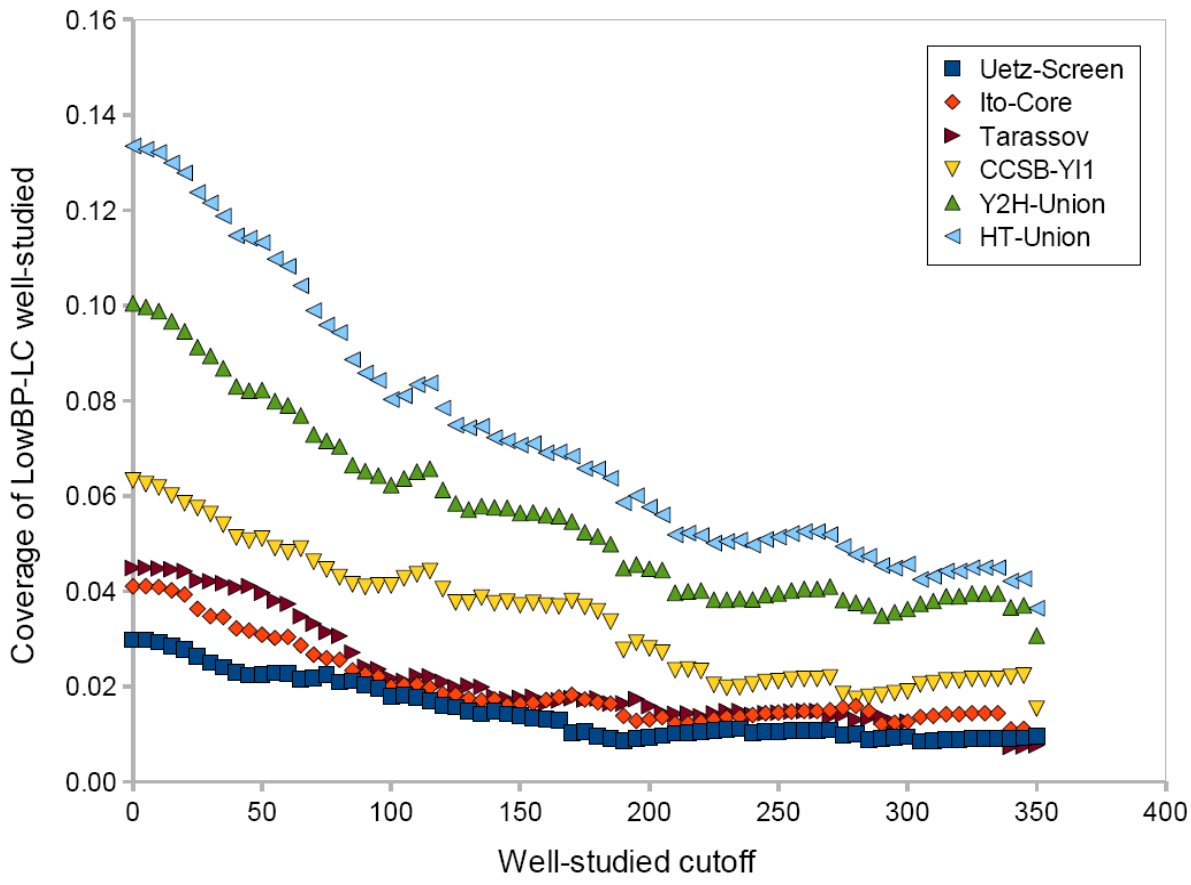

Figure 6: Couverture par divers jeux de données haut débit des interactions issues de la littérature (LowBP-LC) pour les protéines « très étudiées ». La proportion d'interactions de LowBP-LC très étudiées qui sont couvertes par chaque jeux de données haut débit est représentée, en fonction du seuil utilisé pour définir « très étudiée »: il s'agit du nombre minimal d'articles référençant une protéine pour que celle-ci soit considérée comme très étudiée. On observe qu'être plus exigeant pour considérer une protéine comme très étudiée diminue la couverture par les données haut débit.

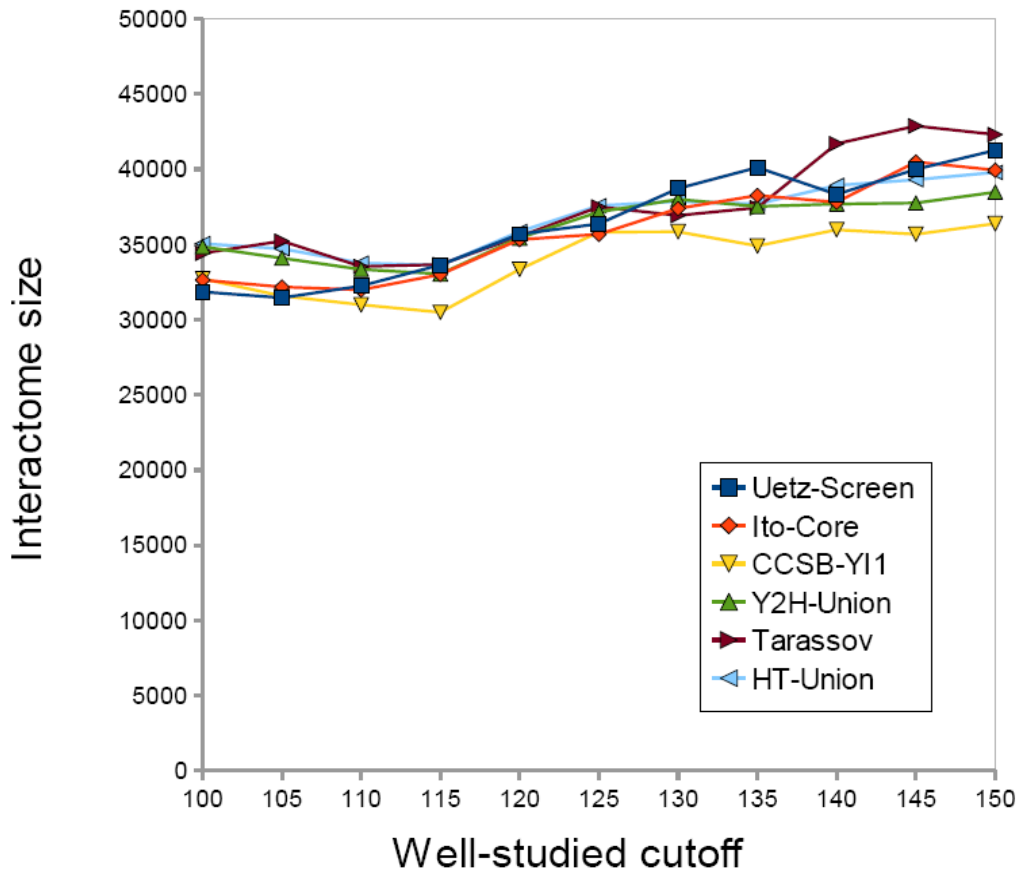

Figure 7: Taille estimée de l'interactome de la levure. Le nombre d'interactions physiques binaires estimé par notre méthode est représenté en fonction du seuil utilisé pour définir les protéines « très étudiées », en utilisant séparément ou conjointement chaque jeu de données haut débit. Les estimations varient peu en fonction du seuil choisi, et même en changeant le jeu de données haut débit.

qu'a reçue chaque protéine de la part de la communauté scientifique, modélisée par le nombre d'articles qui font référence à chaque protéine. Nous avons découvert que l'ensemble des interactions protéine-protéine présentes dans les bases de données de curation de la littérature est qualitativement différent lorsqu'on se restreint aux protéines qui ont été très étudiées. En particulier, ces interactions ont été moins souvent mises en évidence par la méthode double hybride (13.9% de baisse, passe de 58.6% à 44.7%), et plus souvent par des expériences plus lourdes et complexes à mettre en œuvre comme les tests d'activité biochimique (12.4% d'augmentation, passe de 11.1% à 23.5%) ou encore les expériences *in vitro* avec des protéines purifiées (8.5% d'augmentation, passe de 33.5% à 42%). Notre analyse a montré que les jeux de données issues de la littérature et ceux produits à haut débit sont plus corrélées que ce qu'on supposait précédemment, mais que ce biais peut être réduit en se limitant aux protéines intensément étudiées, comme illustré **Figure 6**. Nous avons alors pu proposer une méthode simple et fiable pour estimer la taille

d'un interactome, en combinant les données issues de la littérature concernant les protéines très étudiées avec les données haut débit publiquement disponibles. Cette méthode est robuste aux changements de jeux de données, même si l'on utilise des données produites par des méthodes très différentes (Y2H et Protein Complementation Assay) ainsi qu'aux choix des valeurs des paramètres (**Figure 7**). Au final nous estimons que l'interactome de *S. cerevisiae* comprend au moins 37600 interactions physiques directes binaires.

5. Sujets périphériques: inflammation et insulino-resistance, naissance *de novo* des gènes

J'ai également participé de manière plus ponctuelle à une étude des relations entre inflammation et insulino-résistance chez la souris, en collaboration avec le laboratoire de Goran Hansson au Karolinska Institute (Stockholm). L'obésité est associée à l'inflammation chronique des tissus adipeux. Il a été proposé que des cytokines proinflammatoires, y compris TNFα et IL-6, sécrétées par les tissus adipeux dans le syndrome métabolique, seraient à l'origine d'une insulino-dépendance et favoriseraient le développement du diabète. A l'aide d'une souris mutante *Apoe*$^{-/-}$xCD4dnTGFbR, nous avons montré que l'interleukin-6 est nécessaire au développement de l'insulino-résistance dans les tissus adipeux inflammés, alors que des cytokines “proximales” telles que TNFα ne suffisent pas pour la provoquer.

Cette étude a reposé initialement sur des données d'expression, produites au Karolinska sur une plateforme Affymetrix. Il était nécessaire d'analyser ces données et surtout de les comparer avec celles obtenues dans plusieurs études antérieures, utilisant généralement des plateformes ou des versions de puces différentes: c'est ma contribution à ce projet, modeste mais essentielle car c'est ce qui a permis de pointer du doigt IL-6. Ces résultats ont été publiés en 2009 (Sultan et al, Circ Res 2009).

Un autre projet auquel j'ai participé récemment consiste à étudier les gènes peu conservés chez la levure, avec Anne-Ruxandra Carvunis (CCSB, Boston et TIMC-IMAG, Grenoble), doctorante que j'ai co-encadrée avec Marc Vidal (CCSB, Boston) de 2006 à 2011. Suite au séquençage de plusieurs levures “cousines” de *S. cerevisiae*, il a été proposé que des centaines de gènes dont la séquence ne semblait pas conservée étaient en fait des erreurs de prédiction. Nous nous sommes intéressés à ces gènes mal conservés, dont nous doutions qu'ils soient tous artéfactuels. En effet, une analyse bioinformatique a révélé que nombre d'entre eux ressortent dans diverses expériences haut débit, suggérant qu'ils existent et sont fonctionnels, et des expériences de RT-PCR effectuées au CCSB ont montré l'existence de transcrits pour une proportion significative d'entre eux. Partant de ce constat, nous avons développé un modèle pour expliquer la naissance de ces gènes. Puisqu'ils sont peu ou pas conservés, ils n'ont a priori pas pu apparaître par le classique mécanisme de duplication-divergence. Nous avons donc proposé que le mécanisme de naissance *de novo* des gènes, documenté dans quelques cas particuliers, est en fait largement employé au cours de l'évolution. Selon ce modèle, une séquence intergénique transcrite puis traduite de manière plus ou moins fortuite acquiert occasionnellement une fonction bénéfique et est donc séléctionnée, pour aboutir à la naissance d'un gène. Il existerait ainsi un continuum allant des séquences intergéniques jusqu'aux gènes, en passant par des « proto-gènes » transitoires dont certains deviennent des

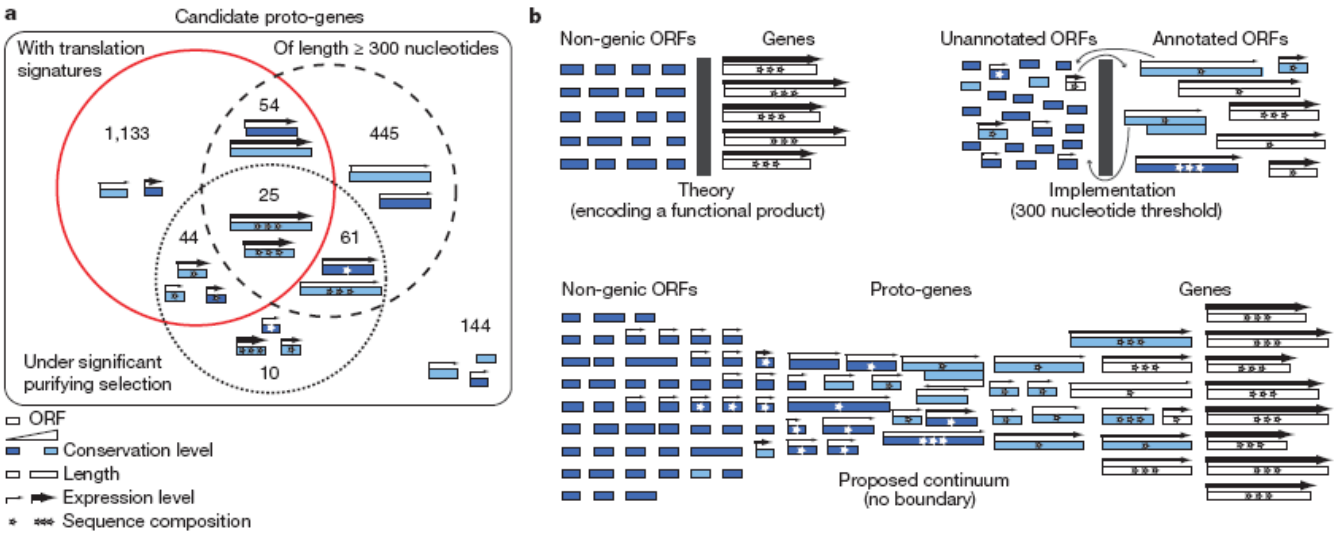

Figure 8: Représentation des proto-gènes dans un continuum allant des séquences intergéniques aux gènes. a. Caractérisation des proto-gènes candidats identifiés comme étant sous sélection purifiante, particulièrement longs, et/ou pour lesquels une signature de traduction a été observée par ribosome profiling. b. Le modèle dichotomique d'annotation des gènes utilisé par la communauté (haut), et le modèle de continuum proposé (bas). Le niveau de conservation, la longueur de l'ORF, et le niveau d'expression des proto-gènes candidats augmentent de manière corrélée et sont finalement similaires à ceux des gènes consensuels pour certains proto-gènes.

gènes à part entière au cours de l'évolution, comme représenté **Figure 8**. Cette hypothèse est supportée par nos analyses bioinformatiques qui ont porté sur une grande diversité de données, allant de l'analyse des séquences jusqu'aux résultats d'expériences de « ribosome footprinting » (qui identifient des séquences traduites). Ces résultats ont été récemment publiés dans Nature (Carvunis et al, Nature 2012). Ma contribution a consisté à superviser et encadrer les travaux d'Anne-Ruxandra, en particulier sur les aspects méthodologiques (statistiques et bioinformatiques).

6. Conclusion et perspectives

J'ai déjà évoqué mes projets pour la suite concernant les deux principaux axes sur lesquels j'ai travaillé depuis une dizaine d'années, à savoir le smart-pooling bien sûr mais aussi de manière moins individuelle l'interactome de la matrice extra-cellulaire. Je garde une activité dans ces deux thématiques à travers les perspectives mentionnées, mais je pense en avoir un peu « fait le tour » et avoir apporté l'essentiel de ce que je devais apporter. J'envisage donc mes travaux futurs sur ces thèmes comme une activité qui deviendra de plus en plus périphérique pour moi, et je me tourne progressivement vers une nouvelle problématique qui devient pervasive en biologie: l'analyse de données de séquençage seconde génération. Je travaille ainsi depuis 2010-2011 sur plusieurs projets de ce type.

Tout d'abord, Laure Sambourg poursuit sa thèse en étudiant des données de séquençage du cancer de l'ovaire, produites par le consortium TCGA: ré-analyse des données brutes comportant notamment alignement et appel des SNVs, intégration avec des données interactome, et prise en compte de l'épissage alternatif, en collaboration avec Gary Bader (U. of Toronto). Elle a passé 6 mois à Toronto en 2011 dans le laboratoire de Gary Bader, pour apprivoiser les données et débuter les analyses, et poursuit son travail au sein de l'équipe BCM depuis son retour.

Un second sujet sur lequel je me suis engagé consiste à étudier des données d'interaction ADN-protéine produites par la méthode ChIP-seq chez *Arabidopsis thaliana*. Ce projet est mené en collaboration avec François Parcy (iRTSV, Grenoble), qui s'intéresse au développement floral chez *Arabidopsis* et en particulier au facteur de transcription *LEAFY*.

Enfin, un troisième projet dans lequel je me suis récemment lancé concerne l'infertilité masculine: en collaboration avec Pierre Ray (IBP, CHU de Grenoble), nous cherchons à identifier des gènes impliqués dans plusieurs formes d'infertilité. Pour ce faire, nous analysons des données exome-seq originales de patients dont les phénotypes ont été soigneusement caractérisés: en partant d'une population porteuse d'une anomalie rare des spermatozoïdes et phénotypiquement homogène, on espère pouvoir trouver les gènes causaux plus facilement. Un premier article issu de cette collaboration a été soumis.

Ces projets sont assez récents et n'ont pas encore fait l'objet de publications, aussi je ne les évoque que rapidement ici. J'y consacre néanmoins d'ores et déjà la majorité de mon temps, et je prévois que mon thème principal de recherches pour les années à venir sera constitué de ce type de projets. En effet l'analyse de données NGS devient une étape clé pour apporter des réponses dans une grande variété de problématiques passionnantes, allant de l'identification de gènes responsables de pathologies diverses à l'élucidation fine des mécanismes de régulation de l'expression. J'espère pouvoir apporter une contribution sur ces problèmes dans les années qui viennent.

J'ai eu la chance jusqu'à présent de trouver des collègues HDR bienveillants, qui acceptaient d'assurer l'encadrement officiel de mes étudiants sans imposer de contraintes, et je les en remercie. Néanmoins, l'obtention de l'Habilitation à Diriger des Recherches apportera une indépendance bienvenue, et me permettra d'avancer dans les meilleures conditions.

Annexe A1: Curriculum Vitae détaillé

A1.1 Coordonnées

Statut: Chargé de Recherches 1ère classe CNRS (depuis 2006),
section 21: Organisation, expression et évolution des génomes, Bioinformatique et biologie des systèmes.

Affectation: Laboratoire TIMC-IMAG (Techniques de l'Ingénierie Médicale et de la Complexité - Informatique, Mathématiques et Applications de Grenoble),
Equipe BCM (Biologie Computationnelle et Mathématique).

Adresse: Laboratoire TIMC-IMAG / BCM, CNRS UMR 5525
Pavillon Taillefer, Faculté de Médecine, 38706 La Tronche cedex, France.

e-mail: Nicolas.Thierry-Mieg@imag.fr

tél: +33/0 456.520.067

fax: +33/0 456.520.055

web: http://www-timc.imag.fr/Nicolas.Thierry-Mieg/

A1.2 Etudes et parcours professionnel

Etudes

• 1991-1994 : Classes préparatoires aux grandes écoles, Lycée Joffre (Montpellier).

• 1994-1997 : Diplôme d'ingénieur en informatique de l'ENSIMAG (Ecole Nationale Supérieure d'Informatique et de Mathématiques Appliquées de Grenoble).

• 1994-1996 : Licence de Mathématiques à l'Université Joseph Fourier (Grenoble 1), préparée et obtenue en parallèle avec l'ENSIMAG.

• 1996-1997 : DEA "Informatique : Systèmes et Communications", Université Joseph Fourier.

• 1997-2001 : Doctorat en (bio-)informatique à l'Université Joseph Fourier.

Titre: Modélisation informatique et analyse prédictive des interactions protéine-protéine chez *Caenorhabditis elegans.*

Travail réalisé conjointement au laboratoire LSR (Laurent Trilling, St-Martin-d'Hères), et au Massachusetts General Hospital puis au Dana Farber Cancer Institute (Marc Vidal, Boston, USA).

Financement:

- Allocation ministérielle (1997-2000),
- Assistant Temporaire d'Enseignement et de Recherches à l'ENSIMAG (2000-2001);
- Bourse complémentaire EURODOC (mobilité internationale des doctorants) de la région Rhône-Alpes (1997-2001).

Parcours professionnel

• 2001 : Recruté Chargé de Recherches 2ème classe au CNRS, section 21: Bases moléculaires et structurales des fonctions du vivant.
Affecté au laboratoire LSR (Logiciels-Systèmes-Réseaux), Saint-Martin-d'Hères.

• 2006 : Promotion au grade de Chargé de Recherches 1ère classe.

• 2007 : Changement d'affectation - je rejoins le laboratoire TIMC-IMAG à La Tronche.

• 2012 : Changement de section - je rejoins la nouvelle section 21: Organisation, expression et évolution des génomes, Bioinformatique et biologie des systèmes.

A1.3 Enseignement et encadrement

Enseignements

2000-2001: • ATER à l'Ecole Nationale Supérieure d'Informatique et de Mathématiques Appliquées de Grenoble (ENSIMAG):
- Algorithmique (42h) en 1ère année (~ L3),
- Réseaux (18h) en 1ère année,
- Systèmes d'exploitation (12h) en 1ère année.

• Interventions dans des modules de bioinformatique (18h):
- M1 et M2/DEA à l'Université Joseph Fourier Grenoble (UJF),
- M1 à l'Ecole Normale Supérieure de Lyon (ENSL).

2001-2004: "Informatique pour le génome" (21h/an) 3ème année (~ M2) à l'ENSIMAG.

2004-2008: "Conception de bases de connaissances et de données en biologie" (18h/an), M2 en Master Ingénieries pour la Santé et le Médicament (UJF).

2007-2010: Intervention dans le module "Complex Systems - Systems Biology", en M2 du master Biosciences à l'ENS Lyon (2h/an, sauf 2008-2009 où le module n'a pas été ouvert).

2008-2010: "Algorithmique et statistique pour la bioinformatique" (18h/an), M2 en Master Ingénieries pour la Santé et le Médicament (UJF).

2010-2011: "Bioinformatique" (54h), 3ème année (~ M2) à l'ENSIMAG et M2 en Master Ingénieries pour la Santé et le Médicament (UJF).

2010-2011: "Interactome networks: construction, visualization and analysis." Intervenant invité (16h sur deux jours), Third International Bioinformatics Software School (IBSS 2011), Avril 2011, Tanger, Maroc.

2011-2012: "Projet de programmation en C" partie pratique (20h), 1ère année (~ L3) à l'ENSIMAG.

2011-2013: "Bioinformatique et Biologie Systémique" (27h), 3ème année (~ M2) à l'ENSIMAG et M2 en Master Ingénieries pour la Santé et le Médicament (UJF).

2012-2013: "Projet de programmation en C" parties théoriques et pratiques (50h), 1ère année (~ L3) à l'ENSIMAG.

Encadrement de stagiaires M1, M2, ingénieurs

1999-2000: Co-encadrement (50%) de F. Boyer, DEA Informatique Systèmes et Communications: "Recherche de signaux de polyadénylation chez *C. elegans*".

2002-2003: Encadrement (100%) de T. Perrissin, DESS Compétence Complémentaire en Informatique: "programmation d'un simulateur pour l'étude de systèmes de codage haut débit".

2003-2004: Encadrement (100%) de G. Bailly, Magistère 2 d'Informatique et PolyTech'Grenoble (M1): "un décodeur déterministe efficace pour le système STD".
La contribution de Gilles au développement de l'algorithme *interpool* lui a valu d'etre co-auteur d'un article: Thierry-Mieg N, Bailly G. Interpool: interpreting smart-pooling results. **Bioinformatics**. 2008 Mar 1; 24(5):696-703.

2003-2004: Co-encadrement (25%) de J. Safon, M2 Master Cryptologie, Sécurité et Codage de l'Information (avec J.-L. Roch, ID Grenoble): "parallélisation et certification de la simulation de criblages haut débit".

2005-2006: Encadrement (100%) de N. Bortolussi, M2 Master Ingénieries pour la Santé et le

Médicament: “Conception et implémentation d’algorithmes pour l'interprétation d'expériences de pooling avec scores discrets”.

2005-2006: Co-encadrement (30%) de E. Chautard, PFE ingénieur en Génie Biologique (~M2), PolyTech'Clermont-Ferrand (avec S. Ricard-Blum, IBCP Lyon): “développement de cartes d’interactions extracellulaires”.

2006-2007: Co-encadrement (30%) de E. Chautard, M2 Master Approches Mathématiques et Informatique du Vivant, UCB Lyon 1 (avec S. Ricard-Blum, IBCP, Lyon) : “développement et analyse de cartes d’interactions extracellulaires”.

2008-2009: Encadrement (100%) de L. Sambourg, M2 Master Ingénieries pour la Santé et le Médicament: “Estimating the size of the *S. cerevisiae* interactome”.

05-08/2011: Encadrement pendant 3 mois de D. Duma, doctorante à University of California Riverside, USA (directeur: S. Lombardi), venue travailler avec moi quelques mois: “Pooling designs and Information Theory”.

Encadrement de doctorants

2006-2011: Co-directeur (25%) de la thèse d'A.-R. Carvunis (avec M. Vidal, CCSB Boston, et L. Trilling, TIMC-IMAG Grenoble): Des protéines et de leurs interactions aux principes évolutifs des systèmes biologiques.
Participation à l'encadrement: 25%.
Thèse soutenue le 26 Janvier 2011.
Publications communes:
• Carvunis AR, Gomez E, Thierry-Mieg N, Trilling L, Vidal M. Systems biology: from yesterday's concepts to tomorrow's discoveries. **Med Sci** (Paris). 2009 Jun-Jul;25(6-7):578-84.
• Carvunis AR, Rolland T, Wapinski I, Calderwood MA, Yildirim MA, Simonis N, Charloteaux B, Hidalgo CA, Barbette J, Santhanam B, Brar GA, Weissman JS, Regev A, Thierry-Mieg N, Cusick ME, Vidal M. Proto-genes and de novo gene birth. **Nature**. 2012 Jul;487:370-4.
Je précise qu'Anne-Ruxandra a également signé plusieurs articles en tant que (co-)premier auteur au cours de sa thèse (**Genome Res** 2008, **Nat Methods** 2009, et deux **Science** 2011), mais j'ai refusé de co-signer ces articles car j'ai estimé que ma contribution à ces travaux n'était pas significative.
Distinction: Anne-Ruxandra est lauréate de la bourse 2009 "Pour les Femmes et la Science" attribuée par l'UNESCO, L'Oréal France et l'Académie des Sciences.

2007-2010: Co-directeur (30%) de la thèse d'E. Chautard (avec S. Ricard-Blum, IBCP Lyon): Construction et analyse de réseaux d’interactions extracellulaires.
Participation à l'encadrement: 30%.
Thèse soutenue le 21 Septembre 2010.
Publications communes:
• Chautard E, Ballut L, Thierry-Mieg N*, Ricard-Blum S* (*: corresponding authors). MatrixDB, a database focused on extracellular protein-protein and protein-carbohydrate interactions. **Bioinformatics**. 2009 Mar 1;25(5):690-1.
• Chautard E, Thierry-Mieg N, Ricard-Blum S. Interaction networks: From protein functions to drug discovery. A review. **Pathol Biol** (Paris). 2009 Jun;57(4):324-33.
• Chautard E, Thierry-Mieg N, Ricard-Blum S. Interaction networks as a tool to investigate

the mechanisms of aging. **Biogerontology**. 2010 Aug;11(4):463-73.
• Chautard E, Fatoux-Ardore M, Ballut L, Thierry-Mieg N, Ricard-Blum S. MatrixDB, the extracellular matrix interaction database. **Nucleic Acids Res**. 2011 Jan;39(Database issue):D235-40.
Présentations orales co-signées:
• Chautard E, Thierry-Mieg N, Ricard-Blum S. Développement de cartes d'interactions extracellulaires. Best student talk award. Talk by E. Chautard, XVème réunion annuelle de la Société Française des Tissus Conjonctifs (SFTC) - 2ème congrès commun SFTC/JFBTM (Journées Françaises de Biologie des Tissus Minéralisés), mai 2006, Lyon.
• Chautard E, Thierry-Mieg N, Fanchon E, Ricard-Blum S. An extracellular matrix interaction map. Talk by E. Chautard, JOBIM'07, juil 2007, Marseille, France.

2009-...: Co-directeur (95%) de la thèse de L. Sambourg (avec J. Demongeot, TIMC-IMAG): “Interactome du cancer du pancréas : génération et analyse”.
Participation à l'encadrement: 95%.
Financement: allocation ministérielle fléchée.
Publication commune:
• Sambourg L, Thierry-Mieg N* (*: corresponding author). New insights into protein-protein interaction data lead to increased estimates of the S. cerevisiae interactome size. **BMC Bioinformatics**. 2010 Dec 21;11(1):605.
Présentations orales ou posters co-signés:
• Sambourg L, Bader G, Thierry-Mieg N. Identifying driver splice forms in ovarian cancer. Poster by L. Sambourg, Canadian Cancer Research Conference, nov 2011, Toronto, Canada.
• Sambourg L, Thierry-Mieg N. New insights into protein-protein interaction data lead to increased estimates of the S. cerevisiae interactome size. Talk by L. Sambourg, Integrative Post-Genomics IPG'2010, nov 2010, Lyon, France.
• Sambourg L, Thierry-Mieg N. Estimating the size of the S. cerevisiae interactome. Poster by L. Sambourg, JOBIM, sept 2010, Montpellier, France.

A1.4 Activités d'intérêt collectif

Travail d'expertise

• Reviewer pour les revues scientifiques:
- Nature Methods,
- Bioinformatics,
- BMC Bioinformatics,
- Nucleic Acids Research,
- Current Opinions in Drug Discovery and Development.

• Expertise pour l'ANR.

• Membre du comité de programme des conférences JOBIM'07 et JOBIM'08.

• Membre du comité de sélection pour un poste MCF section 64 à l'Université Claude Bernard (Lyon), Mai 2011. Intitulé: réseaux d'interactions hôte-pathogènes.

• Membre de jurys de thèses:
- Emilie Chautard, 21 Septembre 2010;
- Anne-Ruxandra Carvunis, 26 Janvier 2011.

Animation Scientifique

2005-2007: Organisation et animation du journal club de l'équipe PLIAGE, laboratoire LSR.

2008-...: Organisation et animation du journal club de l'équipe TIMB puis BCM, laboratoire TIMC-IMAG.

2009-...: Organisation et animation du lab meeting de l'équipe TIMB puis BCM, laboratoire TIMC-IMAG.

2006-...: Membre du comité de pilotage de SeMoVi (Séminaire Rhône-Alpin de Modélisation du Vivant) depuis sa création en 2006.

2008: Co-organisateur (avec A. Schliep, MPI Molecular Genetics, Berlin, et A. Shokrollahi, EPFL Lausanne) du Dagstuhl Seminar "Group Testing in the Life Sciences", Dagstuhl, Allemagne, juillet 2008.

2010: Co-organisateur (avec C. Brun, TAGC Marseille, et J. Reboul, CRCM Marseille) d'un Atelier INSERM "Interactomique: à l'intersection entre biologie et bioinformatique", Saint-Raphaël, printemps 2010.

2010: Intervenant invité à la journée "Débuter une carrière en bioinformatique" organisée par l'association JeBiF (Jeunes Bioinformaticiens de France), journée satellite JOBIM 2010, 6 septembre 2010, Montpellier.

Administration

2003-2007: Membre nommé du conseil de l'unité, laboratoire LSR.

A1.5 Production et communications scientifiques

Publications dans des revues internationales

18. Carvunis AR, Rolland T, Wapinski I, Calderwood MA, Yildirim MA, Simonis N, Charloteaux B, Hidalgo CA, Barbette J, Santhanam B, Brar GA, Weissman JS, Regev A, Thierry-Mieg N, Cusick ME, Vidal M. Proto-genes and de novo gene birth. **Nature**. 2012 Jul;487:370-4.

17. Chautard E, Fatoux-Ardore M, Ballut L, Thierry-Mieg N, Ricard-Blum S. MatrixDB, the extracellular matrix interaction database. **Nucleic Acids Res**. 2011 Jan;39(Database issue):D235-40.

16. Sambourg L, Thierry-Mieg N* (*: corresponding author). New insights into protein-protein interaction data lead to increased estimates of the S. cerevisiae interactome size. **BMC Bioinformatics**. 2010 Dec 21;11(1):605.

15. Chautard E, Thierry-Mieg N, Ricard-Blum S. Interaction networks as a tool to investigate the mechanisms of aging. **Biogerontology**. 2010 Aug;11(4):463-73.

14. Carvunis AR, Gomez E, Thierry-Mieg N, Trilling L, Vidal M. Systems biology: from yesterday's concepts to tomorrow's discoveries. **Med Sci** (Paris). 2009 Jun-Jul;25(6-7):578-84.

13. Xin X*, Rual JF, Hirozane-Kishikawa T, Hill DE, Vidal M+, Boone C+, Thierry-Mieg N+* (*: contributed equally; +: corresponding authors). Shifted Transversal Design smart-pooling for high coverage interactome mapping. **Genome Res**. 2009 Jul;19(7):1262-9.

12. Sultan A, Strodthoff D, Robertson AK, Paulsson-Berne G, Fauconnier J, Parini P, Ryden M, Thierry-Mieg N, Johansson ME, Chibalin AV, Zierath JR, Arner P, Hansson GK. T cell-mediated inflammation in adipose tissue does not cause insulin resistance in hyperlipidemic mice. **Circ Res**. 2009 Apr 24;104(8):961-8.

11. Chautard E, Ballut L, Thierry-Mieg N*, Ricard-Blum S* (*: corresponding authors). MatrixDB, a database focused on extracellular protein-protein and protein-carbohydrate interactions. **Bioinformatics**. 2009 Mar 1;25(5):690-1.

10. Chautard E, Thierry-Mieg N, Ricard-Blum S. Interaction networks: From protein functions to drug discovery. A review. **Pathol Biol** (Paris). 2009 Jun;57(4):324-33.

9. Schliep A, Shokrollahi A, Thierry-Mieg N (editors). Dagstuhl Seminar 08301 Report: Group Testing in the Life Sciences. **Dagstuhl Seminar Proceedings**, July 2008, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany.

8. Thierry-Mieg N*, Bailly G (*: corresponding author). Interpool: interpreting smart-pooling results. **Bioinformatics**. 2008 Mar 1; 24(5):696-703.

7. Thierry-Mieg N. Pooling in systems biology becomes smart. **Nat Methods**. 2006 Mar; 3(3):161-2.

6. Thierry-Mieg N. A new pooling strategy for high-throughput screening: the Shifted Transversal Design. **BMC Bioinformatics**. 2006 Jan 19; 7:28.

5. Davy A, Bello P, Thierry-Mieg N, Vaglio P, Hitti J, Doucette-Stamm L, Thierry-Mieg D, Reboul J, Boulton S, Walhout AJ, Coux O, Vidal M. A protein-protein interaction map of the *Caenorhabditis elegans* 26S proteasome. **EMBO Rep**. 2001 Sep; 2(9):821-8.

4. Reboul J, Vaglio P, Tzellas N, Thierry-Mieg N, Moore T, Jackson C, Shin-i T, Kohara Y, Thierry-Mieg D, Thierry-Mieg J, Lee H, Hitti J, Doucette-Stamm L, Hartley JL, Temple GF, Brasch MA, Vandenhaute J, Lamesch PE, Hill DE, Vidal M. Open-reading-frame sequence tags (OSTs) support the existence of at least 17,300 genes in *C. elegans*. **Nat Genet**. 2001 Mar; 27(3):332-6.

3. Thierry-Mieg N*, Trilling L* (*: corresponding authors). InterDB, a Prediction-Oriented Protein

Interaction Database for *C. elegans*. In O. Gascuel M.F. Sagot eds., JOBIM 2000 selected papers, **Lecture Notes in Computer Science** (2001), 2066, 135-146.

2. Walhout AJ, Sordella R, Lu X, Hartley JL, Temple GF, Brasch MA, Thierry-Mieg N, Vidal M. Protein interaction mapping in *C. elegans* using proteins involved in vulval development. **Science**. 2000 Jan 7; 287(5450):116-22.

1. Walhout M, Endoh H, Thierry-Mieg N, Wong W, Vidal M. A model of elegance. **Am J Hum Genet**. 1998 Oct; 63(4):955-61.

Chapitres d'ouvrages

1. Xin X, Boone C, Thierry-Mieg N. Mapping interactomes with high coverage and efficiency using the shifted transversal design. **Methods Mol Biol**. 2012;812:147-59.

Présentations orales dans des conférences internationales, séminaires invités à l'étranger

14. Thierry-Mieg N. Combinatorial group testing with the Shifted Transversal Design, applications to biology. Talk, Nagoya University, Department of Computer Science and Mathematical Informatics Seminar (host: Masakazu Jimbo), July 2012, Nagoya, Japan.

13. Thierry-Mieg N. Mapping protein-protein interactions with the Shifted Transversal Design. Talk, the second Institute of Mathematical Statistics - Asia Pacific Rim Meeting (IMS-APRM2012), TCP12, July 2012, Tsukuba, Japan.

12. Thierry-Mieg N. Shifted Transversal Design smart-pooling for high-throughput biology. Talk, Institute for Mathematics and Applications (IMA) Workshop on Group Testing Designs, Algorithms, and Applications to Biology. Feb 2012, Minneapolis, USA.

11. Thierry-Mieg N. Interactome networks: construction, visualization and analysis. Intervenant invité (deux jours de cours) à la Third International Bioinformatics Software School (IBSS 2011), Avril 2011, Tanger, Maroc.

10. Thierry-Mieg N. Smart Pooling: increasing accuracy, coverage and efficiency in high-throughput screening. Talk, Ontario Institute for Cancer Research (host: Francis Ouellette), Sept 2008, Toronto, Canada.

9. Thierry-Mieg N. Smart Pooling: increasing accuracy, coverage and efficiency in high-throughput screening. Talk, University of Toronto Department of Molecular Genetics Guest Speaker Series, Sept 2008, Toronto, Canada.

8. Xin X*, Thierry-Mieg N*, Rual JF, Hirozane-Kishikawa T, Hill D, Vidal M, Boone C (*: contributed equally). Smart-pooling for proteome-scale interactome mapping using the Shifted Transversal Design. Talk, 2008 CSHL/WT Network Biology Meeting, Aug 2008, Hinxton, UK.

7. Thierry-Mieg N. Smart-pooling. Talk, Dagstuhl Seminar: Group Testing in the Life Sciences, Jul 2008, Dagstuhl, Germany.

6. Thierry-Mieg N. A new smart-pooling strategy for high-throughput screening: the Shifted Transversal Design. Talk, DIMACS Workshop on Combinatorial Group Testing, May 2006, Rutgers U., New Jersey, USA.

5. Thierry-Mieg N. A new pooling strategy for high-throughput screening: The Shifted Transversal Design. Flash talk, JOBIM'05, Jul 2005, Lyon, France.

4. Thierry-Mieg N. Pooling strategies for interactome mapping. Center for Cancer Systems Biology (host:

Marc Vidal), April 2002, Boston, USA.

3. Thierry-Mieg N. Predicting protein-protein interactions. Center for Cancer Systems Biology (host: Marc Vidal), April 2002, Boston, USA.

2. Thierry-Mieg N. Protein-protein interaction prediction for *C. elegans* (talk). Knowledge discovery in biology workshop, 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD), Sept 2000, Lyon, France.

1. Thierry-Mieg N. InterDB, a prediction-oriented protein interaction database for *C. elegans* (talk). JOBIM 2000, Journées Ouvertes Biologie Informatique Mathématiques, May 2000, Montpellier.

Présentations orales dans des conférences nationales, séminaires invités en France

9. Thierry-Mieg N. Smart-pooling: increasing accuracy, coverage and efficiency in high-throughput screening. Talk, MAGMA Seminar, June 2008, Marseille, France.

8. Thierry-Mieg N. Smart-pooling for interactome mapping. Talk, Towards Systems Biology workshop, Oct 2007, Grenoble, France.

7. Thierry-Mieg N. The Shifted Transversal Design. Séminaire à l'Institut des Hautes Etudes Scientifiques (IHES, invitation: Prof. M. Gromov), 22 juin 2004, Bures-sur-Yvette.

6. Thierry-Mieg N. Une nouvelle méthode de construction de pools pour le criblage haut-débit : le Shifted Transversal Design. Séminaire au laboratoire TIMC-IMAG, 28 avril 2004, Grenoble.

5. Thierry-Mieg N. Prédiction d'interactions protéine-protéine par fouille de données (talk). Journée IMPG: Protéomique et bioinformatique structurales et fonctionnelles. 20-21 janvier 2003, Marseille-ESIL Luminy.

4. Thierry-Mieg N. Conceptual modeling and predictive analysis of protein-protein interactions in *Caenorhabditis elegans* (talk). Journées de Post Génomique de la Doua JPGD'02, mars 2002, Lyon.

3. Thierry-Mieg N. KDD et prédiction d'interactions protéine-protéine. Séminaire à l'Equipe Universitaire de Recherche en Informatique de Saint-Etienne (EURISE), février 2002, Saint-Etienne.

2. Thierry-Mieg N. Prédiction d'interactions protéine-protéine (talk). Journée IBIM'02 (Interface Biologie, Informatique et Modélisation). Janvier 2002, Grenoble.

1. Thierry-Mieg N. Les approches de génomique fonctionnelle chez *C. elegans*. Intervenant invité à la deuxième Ecole Thématique de Biologie Végétale CNRS/INRA, génomique fonctionnelle chez les végétaux : du gène à la fonction. Mars 2001, Carry-le-Rouet.

Posters présentés dans des conférences internationales

19. Sambourg L, Bader G, Thierry-Mieg N. Identifying driver splice forms in ovarian cancer. Poster (L. Sambourg), Canadian Cancer Research Conference, nov 2011, Toronto, Canada.

18. Sambourg L, Thierry-Mieg N. New insights into protein-protein interaction data lead to increased estimates of the S. cerevisiae interactome size. Talk by L. Sambourg, Integrative Post-Genomics IPG'2010, nov 2010, Lyon, France.

17. Sambourg L, Thierry-Mieg N. Estimating the size of the S. cerevisiae interactome. Poster (L. Sambourg), JOBIM, sept 2010, Montpellier, France.

16. Chautard E, Ballut L, Thierry-Mieg N, Ricard-Blum S. From static to dynamic extracellular interaction networks. Poster (E. Chautard), Second Joint Meeting of the French and German Connective Tissue Societies and CARD, June 2009, Reims, France.

15. Xin X*, Thierry-Mieg N*, Hill D, Vidal M, Boone C (*: contributed equally). Smart-pooling for proteome-scale interactome mapping using the Shifted Transversal Design (STD). Poster (X. Xin), 2008 Yeast Genetics and Molecular Biology Meeting, Jul 2008, Toronto, Canada.

14. Chautard E, Faye C, Thierry-Mieg N, Fanchon E, Ricard-Blum S. The first draft of the human extracellular interaction network: how do extracellular molecules work together? Talk (E. Chautard) in Vascular Biology and Angiogenesis workshop, XXIst FECTS meeting (Federation of European Connective Tissue Societies), Jul 2008, Marseille, France.

13. Ballut L, Chaboud A, Grosjean I, Braun P, Vidal M, Thierry-Mieg N, Ricard-Blum S. Validation of a protein interaction gold standard by surface plasmon resonance arrays. Poster, XXIst FECTS meeting (Federation of European Connective Tissue Societies), Jul 2008, Marseille, France.

12. Chautard E, Faye C, Thierry-Mieg N, Ricard-Blum S. MatrixDB: an extracellular matrix interactions database. Poster, April 2008, Barcelona BioMed Conference on Targeting and Tinkering with Interaction Networks, Barcelona, Spain.

11. Thierry-Mieg N, Rual J-F, Hirozane-Kishikawa T, Hill D, Vidal M. Smart pooling for sensitive and specific interactome mapping: a pilot experiment (poster). 2007 CSHL/WT Interactome Networks Meeting, Aug 2007, Hinxton, UK.

10. Chautard E, Thierry-Mieg N, Fanchon E, Ricard-Blum S. An extracellular matrix interaction network (poster). 2007 CSHL/WT Interactome Networks Meeting, Aug 2007, Hinxton, UK.

9. Chautard E, Fanchon E, Thierry-Mieg N, Ricard-Blum S. An extracellular matrix interaction map (poster). Gordon Research Conference Collagen, July 2007, New London, USA.

8. Chautard E, Thierry-Mieg N, Ricard-Blum S. A first step towards the extracellular matrix interactome (poster). Integrative Post Genomics (IPG) nov. 2006, Lyon.

7. Chautard E, Thierry-Mieg N, Ricard-Blum S. A first step towards the extracellular matrix interactome (poster). XXth FECTS meeting (Federation of European Connective Tissue Societies), Jul 2006, Oulu, Finland.

6. Thierry-Mieg N. Pooling for the interactome: better, faster, cheaper! (poster). 4th International ORFeome Meeting, Dec 2004, Boston, USA.

5. Thierry-Mieg N, Trilling L, Roch JL. A “smart pooling” system for protein-protein interaction mapping (poster). European Conference on Computational Biology (ECCB) 2003, Sept 2003, Paris, France.

4. Thierry-Mieg N. Predicting protein-protein interactions for the *C. elegans* interactome project (poster). European Conference on Computational Biology (ECCB) 2002, Oct 2002, Saarbrücken, Germany.

3. Walhout M, Sordella R, Thierry-Mieg N, Brasch M, Temple G, Hartley J, Lorson M, van den Heuvel S, Endoh H, Vidal M. The *C. elegans* protein interaction mapping project : a test-case using proteins involved in vulval development. 12th International *C. elegans* Meeting, June 1999, University of Wisconsin, Madison, USA.

2. Vidal M, Walhout M, Sordella R, Thierry-Mieg N, Brasch M, Temple G, Hartley J, Lorson M, van den Heuvel S, Endoh H. The *C. elegans* protein interaction mapping project : a test-case using proteins involved in vulval development. 12th Annual Meeting on Genome Mapping, Sequencing & Biology, May 1999, Cold Spring Harbour Laboratory, USA.

1. Vidal M, Endoh H, Thierry-Mieg N, Walhout M, Wong W. Description of a protein-protein interaction

mapping project. 1998 East Coast *C. elegans* Meeting, June 1998, Boston University, USA.

Logiciels

1. Conception et développement du logiciel Interpool. Ce logiciel remplit les rôles suivants: construction des pools du Shifted Transveral Design; interprétation des résultats d'expériences de pooling; simulation d'expériences de criblages de pools (ce qui permet par exemple de choisir les valeurs optimales des paramètres de STD); calcul de l'entropie du système STD. Le logiciel comporte environ 18000 lignes de code en C. Il est distribué sous licence GNU GPL. http://www-timc.imag.fr/Nicolas.Thierry-Mieg/smartPooling.html

A1.6 Projets subventionnés

2003-2004: Porteur, projet BQR INPG “Systèmes de codage pour le criblage haut-débit” (avec J.-L. Roch, ID Grenoble, 16k€ dont 12k€ LSR).

2006-2008: Co-porteur, projet CIBLE Région Rhône-Alpes “Les interactomes des domaines de collagène : une clé pour leurs fonctions?” (avec S. Ricard-Blum, IBCP Lyon, 39k€ dont 10k€ TIMC-IMAG).

2008-2009: Partenaire, projet de l'Institut des Systèmes Complexes Rhône-Alpin (IXXI) “Biologie systémique de l'angiogenèse : construction et analyse de modèles discrets de réseaux d'interaction ” (avec E. Fanchon, TIMC-IMAG, et S. Ricard-Blum, IBCP Lyon, 5k€ dont 3k€ TIMC-IMAG).

2009-2011: Co-porteur, projet Cluster Infectiologie Région Rhône-Alpes “Perturbations par le virus de l’hépatite C de la voie de signalisation du TGFβ, conséquences pour le développement de la fibrose hépatique” (avec V. Lotteau, IIV Lyon, et S. Ricard-Blum, IBCP Lyon, 30k€ dont 4k€ TIMC-IMAG).

2009-2011: Co-porteur, projet Cluster Infectiologie Région Rhône-Alpes “Interactions des parasites Leishmania avec la matrice extracellulaire. Rôle dans le tropisme tissulaire et l'infectivité” (avec S. Ricard-Blum, IBCP Lyon, et P. Esterre, Institut Pasteur de Cayenne, 36k€ dont 4k€ TIMC-IMAG).

2009-2012: Obtention d'une allocation de recherches ministérielle fléchée: “Interactome du cancer du pancréas : génération et analyse”.

2010-2012: Co-porteur, projet de l'Institut des Systèmes Complexes Rhône-Alpin (IXXI) “Isoformes et spécificité tissulaire dans les réseaux d'interactions extracellulaires” (avec S. Ricard-Blum, IBCP Lyon, 10k€ dont 3k€ TIMC-IMAG).

2011-2012: Co-porteur, projet de l'Institut des Systèmes Complexes Rhône-Alpin (IXXI) “Construction d’un interactome dynamique et quantitatif régulant l'angiogenèse” (avec S. Ricard-Blum, IBCP Lyon, 5k€ dont 2k€ TIMC-IMAG).

2012-2014: Co-porteur, projet ARC 1 Santé Région Rhône-Alpes “Les protéine-kinases de Leishmania : implications dans la virulence et les interactions avec l’hôte” (avec S. Ricard-Blum, IBCP Lyon, 14k€ dont 2.5k€ TIMC-IMAG).

Annexe A2: publications choisies

Je reproduis dans les pages qui suivent mes principales publications sur le thème smart-pooling. Je rappelle cependant que ces articles sont également disponibles comme la plupart de mes publications sur ma page web. Je n'inclus pas ici les suppléments des publications, mais j'encourage le lecteur à regarder en particulier le Supplementary Data de [Xin et al, Genome Res 2009] (qui est aussi téléchargeable sur ma page), dans lequel sont inclus les données brutes, les programmes informatiques, les résultats obtenus, et une documentation qui se veut suffisante pour que nos analyses puissent être intégralement répliquées.

Suivent par ordre chronologique:

Thierry-Mieg N. A new pooling strategy for high-throughput screening: the Shifted Transversal Design. **BMC Bioinformatics**. 2006 Jan 19; 7:28.

Thierry-Mieg N*, Bailly G. Interpool: interpreting smart-pooling results. **Bioinformatics**. 2008 Mar 1; 24(5):696-703.

Xin X*, Rual JF, Hirozane-Kishikawa T, Hill DE, Vidal M+, Boone C+, Thierry-Mieg N+*. Shifted Transversal Design smart-pooling for high coverage interactome mapping. **Genome Res**. 2009 Jul;19(7):1262-9.

BMC Bioinformatics

BioMed Central

Research article

Open Access

A new pooling strategy for high-throughput screening: the Shifted Transversal Design

Nicolas Thierry-Mieg*

Address: Laboratoire Logiciels-Systèmes-Réseaux, IMAG Institute, BP53, 38041 Grenoble Cedex 9, France

Email: Nicolas Thierry-Mieg* - Nicolas.Thierry-Mieg@imag.fr

* Corresponding author

Published: 19 January 2006

*BMC Bioinformatics*2006, **7**:28 doi:10.1186/1471-2105-7-28

Received: 17 June 2005

Accepted: 19 January 2006

This article is available from: http://www.biomedcentral.com/1471-2105/7/28

Abstract

Background: In binary high-throughput screening projects where the goal is the identification of low-frequency events, beyond the obvious issue of efficiency, false positives and false negatives are a major concern. Pooling constitutes a natural solution: it reduces the number of tests, while providing critical duplication of the individual experiments, thereby correcting for experimental noise. The main difficulty consists in designing the pools in a manner that is both efficient and robust: few pools should be necessary to correct the errors and identify the positives, yet the experiment should not be too vulnerable to biological shakiness. For example, some information should still be obtained even if there are slightly more positives or errors than expected. This is known as the group testing problem, or pooling problem.

Results: In this paper, we present a new non-adaptive combinatorial pooling design: the "shifted transversal design" (STD). It relies on arithmetics, and rests on two intuitive ideas: minimizing the co-occurrence of objects, and constructing pools of constant-sized intersections. We prove that it allows unambiguous decoding of noisy experimental observations. This design is highly flexible, and can be tailored to function robustly in a wide range of experimental settings (i.e., numbers of objects, fractions of positives, and expected error-rates). Furthermore, we show that our design compares favorably, in terms of efficiency, to the previously described non-adaptive combinatorial pooling designs.

Conclusion: This method is currently being validated by field-testing in the context of yeast-two-hybrid interactome mapping, in collaboration with Marc Vidal's lab at the Dana Farber Cancer Institute. Many similar projects could benefit from using the Shifted Transversal Design.

Background

With the availability of complete genome sequences, biology has entered a new era. Relying on the sequencing data of genomes, transcriptomes or proteomes, scientists have been developing high-throughput screening assays and undertaking a variety of large scale functional genomics projects. While some projects involve quantitative measurements, others consist in applying a basic yes-or-no test to a large collection of samples or "objects", – be they individuals, clones, cells, drugs, nucleic acid fragments, proteins, peptides... A large class of these binary tests aims at identifying relatively rare events. The main goal is of course to obtain information as efficiently and as reliably as possible. Typically, this is achieved by minimizing the cost of the basic assay in terms of time and money, and automating and parallelizing the experiments as much as

possible. A major difficulty stems from the fact that high-throughput biological assays are usually somewhat noisy: reproducibility is a known problem of microarray analyses, and both false positive and false negative observations are to be expected in binary type experiments. These experimental artifacts should be identified and properly treated. A clean way to deal with the issue consists in repeating all tests several times, but this is usually prohibitively expensive and time-consuming. A more practical approach, in the case of binary tests, consists in retesting all positive results obtained in a first round. This strategy identifies most of the false positives at a reduced cost, but is powerless with regard to false negatives, leaving us in need of a better solution.

In the case of binary experiments testing for rare events, an intuitively appealing strategy consists in pooling the samples to minimize the number of tests. It requires three conditions. First, the objects under scrutiny must be available individually, in a tagged form. For example, a cDNA library in bulk is not exploitable, but a collection of cDNA clones or of cloned coding regions, such as the one produced by the *C. elegans* ORFeome project [1], is fine. Second, it must be possible to test a pool of objects in a single assay and obtain a positive readout if at least one of the objects is positive. For example, this is the case when searching for a specific DNA sequence by PCR in a mixture of molecules: a product will be amplified if at least one of the pooled molecules contains the target sequence. Third, pooling is especially desirable and efficient when the fraction of expected positives is small (at most a few percent). Under these conditions, pooling strategies can be applied, and the difficulty then consists in choosing a "good" set of pools. This being an intuitive but rather vague goal, it must be formalized. A simple formulation, known as the group testing problem (or pooling problem), is the following. Consider a set of n events which can be true or false, represented by n Boolean variables. Let us call "pool" a subset of variables. We define the value of a pool as the disjunction (i.e., the logical OR operator) of the variables that it contains. Let us assume that at most t variables are true. The goal is to build a set of v pools, where v is small compared to n, such that by testing the values of the v pools, one can unambiguously determine the values of the n variables.

If the pools must be specified in a single step, rather than incrementally by building on the results of previous tests, the problem is called "non-adaptive". Although adaptive designs can require fewer tests, non-adaptive pooling designs are often better suited to high-throughput screening projects because they allow parallelization and facilitate automation of the experiments, and also because the same pools can be used for all targets, thereby reducing the total project cost.

The ability to deal with noisy observations is an important added benefit to using a pooling system, compared to the classical individual testing strategy. Indeed, noise detection and correction capabilities are inherent in any pooling system, because each variable is present in several pools, hence tested many times. Depending on the expected noise level, the redundancy can be chosen at will, and simply testing a few more pools than would be necessary in the absence of noise results in robust error-correction. It should be noted that minimization of the number of pools and noise correction are two conflicting goals: increasing noise tolerance generally requires testing more pools. Designing a set of pools requires balancing these two objectives, and finding the right compromise to suit the application.

Other application-dependent constraints may be imposed. In particular, the pool sizes are often limited by the experimental setting. For example, in the context of the *C. elegans* protein interaction mapping project led by Marc Vidal [2,3], it is estimated that, using their high-throughput two-hybrid protocol, reliable readouts can be obtained with pools containing 400 AD-Y clones, or perhaps up to 1000 by tweaking the assay (Marc Vidal, personal communication).

Many groups have used with some success variants of the simple "grid" design, which consists in arraying the objects on a grid and pooling the rows and columns [e.g. [4-6]]. However, although it is better than no pooling, this rudimentary design is vulnerable to noise and behaves poorly when several objects are positive, in addition to being far from optimal in terms of numbers of tests.

In answer to its shortcomings, more sophisticated error-correcting pooling designs have been proposed. Some of these designs are very efficient in terms of numbers of tests, but lack the robustness and flexibility that most real biological applications require. Others are more adaptable and noise-tolerant at the expense of performance. In this paper, we present a new pooling algorithm: the "shifted transversal design" (STD). This design is highly flexible: it can be tailored to allow the identification of any number of positive objects and to deal with important noise levels. Yet it is extremely efficient in terms of number of tests, and we show that it compares favorably to the previously described pooling designs.

The paper is organized as follows. After providing a formal definition STD, we show that it constitutes an error-correcting solution to the pooling problem. The theoretical performance of STD is then evaluated and compared with the main previously described deterministic pooling designs. Finally, we summarize our results and discuss future directions.

Results (1): the Shifted Transversal Design

Preliminaries

The following notations are used throughout this paper, in accordance with the notations from [7].

Let $n \geq 2$, and consider the set $\mathcal{A}_n = \{A_0,...,A_{n-1}\}$ of n Boolean variables.

We will call "pool" a subset of $\mathcal{A}_n$. We say that a pool is "true", or "positive", if at least one of its elements is true.

Let us call "layer" a partition of $\mathcal{A}_n$.

Let q be a prime number, with $q < n$.

We define the "compression power" of q relative to n, noted $\Gamma(q,n)$, as the smallest integer γ such that $q^{\gamma+1} \geq n$. We will simply write Γ for $\Gamma(q,n)$ whenever possible.

Let σ_q be the mapping of $\{0,1\}^q$ onto itself defined by:

$$\forall (x_1,\ldots,x_q) \in \{0,1\}^q, \quad \sigma_q \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_q \end{bmatrix} = \begin{bmatrix} x_q \\ x_1 \\ \vdots \\ x_{q-1} \end{bmatrix}.$$

Note that σ_q is a cyclic function of order q: σ_q^q is the identity function on $\{0,1\}^q$.

The matrix representation

Any set of pools can be represented by a Boolean matrix, as follows. Each column corresponds to one variable, and each row to one pool. The cell (i,j) is true (value 1) if pool i contains variable j, and false (value 0) otherwise.

Example

Consider the n = 9 variables $\mathcal{A}_9 = \{A_0, A_1,...,A_8\}$. The following matrix defines a set of 3 pools:

$$M_0 = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

The pools are $\{A_0,A_3,A_6\}$ (defined by the first row), $\{A_1,A_4,A_7\}$ (second row), and $\{A_2,A_5,A_8\}$ (third row). In fact, this set of pools clearly constitutes a layer.

Definition of STD

A pooling design is a method to construct a set of pools. When the set of pools can be partitioned into layers (i.e. subsets which each form a partition of the set of variables), the pooling design is said to be "transversal". STD is a transversal pooling design that rearranges the variables from one layer to the next, with two intuitive goals in mind. First, the number of pools in which any pair of variables can occur (i.e. the co-occurrence of variables) should be limited: this is essential for determining the variables' values. The second aim is that the intersections between pools should be of roughly constant size, in order to maximize the mutual information obtained by observing the pools' values and thus increase STD's efficiency.

Given a prime number q with $q < n$, and k such that $k \leq q+1$, STD constructs a set STD(n; q; k) of pools composed of k layers. When $k \leq q$, the layers have a uniform construction: they each contain q pools of n/q or (n/q)+1 variables, and are globally interchangeable. In the special case where k = q+1, the q homogeneous layers are supplemented with a singular layer, which has a specific construction and is less regular, yet complements the others nicely. A formal definition of STD(n; q; k) follows.

For every $j \in \{0,...,q\}$, let M_j be a q × n Boolean matrix, defined by its columns $C_{j,0},...,C_{j,n-1}$ as follows:

$$C_{0,0} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \text{ and } \forall\ i \in \{0,...,n-1\} \quad C_{j,i} = \sigma_q^{s(i,j)}(C_{0,0})$$

where:

$$s(i,j) = \sum_{c=0}^{\Gamma} j^c \cdot \left\lfloor \frac{i}{q^c} \right\rfloor \text{ if } j < q, \text{ and } s(i,q) = \left\lfloor \frac{i}{q^\Gamma} \right\rfloor, \text{ where}$$

the semi-bracket denotes the integer part.

Let $L(j)$ be the set of pools of which M_j is the matrix representation. Note that each column $C_{j,i}$ has exactly one occurrence of '1' and (q-1) occurrences of '0'. The index of the '1' identifies the (single) pool of $L(j)$ which contains variable A_i. Therefore, in a given set of pools $L(j)$, each variable is present in exactly one pool, that is to say $L(j)$ constitutes a partition of $\mathcal{A}_n$: $L(j)$ is a layer.

Finally, for $k \in \{1,2,...,q+1\}$, STD(n; q; k) is defined as: $\text{STD}(n; q; k) = \bigcup_{j=0}^{k-1} L(j)$.

Example

Consider again the variables $\mathcal{A}_n$, and let q = 3 (hence Γ = 1). M_0 is as defined above, and M_1, M_2, M_3 are:

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \end{bmatrix},$$

$$M_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \end{bmatrix},$$

$$M_3 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

The corresponding layers of pools are the following:

Layer 0: $L(0) = \{\{A_0,A_3,A_6\}, \{A_1,A_4,A_7\}, \{A_2,A_5,A_8\}\}$

Layer 1: $L(1) = \{\{A_0,A_5,A_7\}, \{A_1,A_3,A_8\}, \{A_2,A_4,A_6\}\}$

Layer2: $L(2) = \{\{A_0,A_4,A_8\}, \{A_1,A_5,A_6\}, \{A_2,A_3,A_7\}\}$

Layer3: $L(3) = \{\{A_0,A_1,A_2\}, \{A_3,A_4,A_5\}, \{A_6,A_7,A_8\}\}$.

STD(9; 3; 2) is the following set of pools: STD(9; 3; 2) = $L(0) \cup L(1)$.

Remark

The method builds at most q+1 layers: indeed, if we discard the last particular layer $L(q)$ and attempt to extend the STD construction to any j, it becomes cyclic of order q: for every j, $L(j+q) = L(j)$.

Results (2): properties of STD

In this section, we establish an important theorem, leading to three corollaries which show that STD constitutes a solution to the pooling problem described in the introduction, and that it can be used to detect and correct noisy observations. We then establish another property of STD, which is noteworthy albeit not directly related to the pooling problem.

Co-occurrence of variables

So far we have considered the variables that are contained in a given pool. Dually, we may consider the set of pools that contain a given variable. For k ∈ {1,2,...,q+1}, we will note $pools_k(i)$ the set of pools of STD(n; q; k) that contain variable A_i:

$\forall$ i ∈ {0,...,n-1}, $pools_k(i) = \{p \in \text{STD}(n; q; k) \mid A_i \in p\}$.

Theorem 1

Recall that q is prime.

$\forall$ i_1, i_2 ∈ {0,...,n-1}, $[i_1 \neq i_2] \Rightarrow [\text{Card}(pools_{q+1}(i_1) \cap pools_{q+1}(i_2)) \leq \Gamma(q,n)]$.

Proof

see Methods section.

Example

Consider again the example n = 9, q = 3, k = 4, for which the layers $L(0)$, $L(1)$, $L(2)$, and $L(3)$ are known (see above). The set of pools containing A_0 is: $pools_4(0) = \{\{A_0,A_3,A_6\},\{A_0,A_5,A_7\},\{A_0,A_4,A_8\},\{A_0,A_1,A_2\}\}$.

One can easily see that A_0 is present exactly once with each other variable. In fact, each pair of variables is present in exactly 1 (= Γ(3,9)) pool, in conformity with theorem 1.

Remark

The property holds a fortiori when k < q+1, i.e. when considering STD(n; q; k) instead of STD(n; q; q+1).

A solution to the pooling problem

Corollary 1

Let t be an integer such that t·Γ(q,n) ≤ q. Let k = t·Γ+1, and consider the set of pools STD(n; q; k). Suppose that the value of each pool has been observed, and that there are at most t positive variables in $\mathcal{A}_n$. Then, in the absence of noise (i.e., if all pool values are correctly observed), the value of every variable can be identified.

Proof

Consider the following algorithm, which tags variables as negative or positive.

Algorithm 1: all the variables present in at least one negative pool are tagged negative; any variable present in at least one positive pool where all other variables have been tagged negative, is tagged positive.

We show that this algorithm correctly identifies the value of each and every variable.

Let A_i be a negative variable. A_i is present in exactly k pools: one pool in each layer. Theorem 1 asserts that no variable other than A_i is present in more than Γ of these t·Γ+1 pools. Therefore, since at most t variables are positive, A_i is present in at least one pool where no positive variable is present. Consequently, examination of this pool yields a negative answer (since all observations are correct), which leads algorithm 1 to tag A_i negative. This shows that every negative variable is correctly tagged as such.

Now let A_i be a positive variable. Since we suppose here that there are no observational errors, all pools containing A_i are positive: A_i is not tagged negative. Again according to theorem 1, no other variable is present in more than Γ of these $t \cdot \Gamma+1$ pools. Therefore, since there are at most t-1 other positive variables, A_i is present in at least $\Gamma+1$ positive pools where all other variables are negative, and are tagged negative according to the above. This shows that every positive variable is tagged correctly and uniquely.

Finally, since every positive pool must contain at least one positive variable, and since no positive variable is tagged negative, we can conclude that no negative variable can be tagged positive (in addition to its negative tag): every negative variable is also uniquely tagged. This completes the corollary's proof.

Example

Consider again our example STD(9; 3; 2) = $\{\{A_0,A_3,A_6\}, \{A_1,A_4,A_7\}, \{A_2,A_5,A_8\}, \{A_0,A_5,A_7\}, \{A_1,A_3,A_8\}, \{A_2,A_4,A_6\}\}$.

Let t = 1, and suppose that a single variable in $\mathcal{A}_9$ is positive. For reasons of symmetry, the name of that variable is inconsequent: all are equivalent. Let us suppose that the only positive variable is A_8. Then pools $\{A_0,A_3,A_6\}$, $\{A_1,A_4,A_7\}$, $\{A_0,A_5,A_7\}$, and $\{A_2,A_4,A_6\}$ are negative, which shows that variables A_0, A_1,...,A_7 are negative; and pools $\{A_2,A_5,A_8\}$ and $\{A_1,A_3,A_8\}$ are positive, which each prove that A_8 is positive (given that A_2, A_5, A_1 and A_3 have been shown to be negative).

Remark

If more than t variables are positive, this fact is revealed: clearly, at most n - (t+1) variables are tagged negative, contrary to when there are at most t positives. In fact, all tags produced by the above algorithm are still correct, but some variables may not be tagged at all: these variables are called "unresolved", or "ambiguous". It would be interesting to know how many ambiguous variables are to be expected, but this is a very hard problem to study analytically, particularly when one takes into account experimental noise. Instead, this issue can be suitably approached by computer simulation.

Dealing with noise: error correction

As stated in the introduction, pooling designs have an intrinsic potential for noise-correction, due to the redundancy of variables. In the case of STD, this potential can be taken advantage of by simply testing a few extra layers of pools and using a modified algorithm, as shown here.

Corollary 2

Let t and E be integers such that $t \cdot \Gamma(q,n)+2 \cdot E \leq q$, and let $k = t \cdot \Gamma+2 \cdot E+1$. Consider the set of pools STD(n; q; k), and suppose that the value of each pool has been observed. Furthermore, suppose that there are at most t positive variables in $\mathcal{A}_n$, and that there are at most E observation errors. Then, all errors can be detected and corrected, and the value of every variable can be identified.

Proof

Consider the following tagging algorithm.

Algorithm 2: all the variables present in at least E+1 negative pools are tagged negative; any variable present in at least E+1 positive pools where all other variables have been tagged negative, is tagged positive.

The proof is similar to that of corollary 1: we show that algorithm 2 correctly and uniquely tags every variable. In this case, theorem 1 shows that each negative variable is necessarily present in at least $2 \cdot E+1$ negative pools. Since there are at most E observation errors, it follows that at least E+1 of these negative pools are correctly observed. Therefore, algorithm 2 tags all negative variables as such. In addition, at most E pools containing a positive variable can be observed negative; hence no positive variable is tagged negative. Finally, since at most E pools containing only negative variables can be observed positive, no negative variable is tagged positive:every negative variable is correctly and uniquely tagged.

Conversely, a positive variable A_i appears in at least $t \cdot \Gamma+E+1$ positive pools (since there are at most E errors), of which at most $(t-1) \cdot \Gamma$ contain at least one other positive variable (according to theorem 1). Therefore A_i is present in at least $(t \cdot \Gamma+E+1) - (t-1) \cdot \Gamma = \Gamma+E+1$ positive pools where all other variables are negative. Since these negative variables have been correctly tagged as such (as shown above), A_i is tagged positive. This shows that algorithm 2 also correctly and uniquely tags all positive variables.

Finally, any observation which is contradictory with the obtained tagging is necessarily erroneous. In other words, false negative and false positive observations are identified.

Remark

Few restrictions are imposed when choosing the value of the parameter q: it must simply be a prime number smaller than n. Consequently, STD can be used successfully even when very high noise levels are expected, by picking a large value for q. Of course, as is to be expected

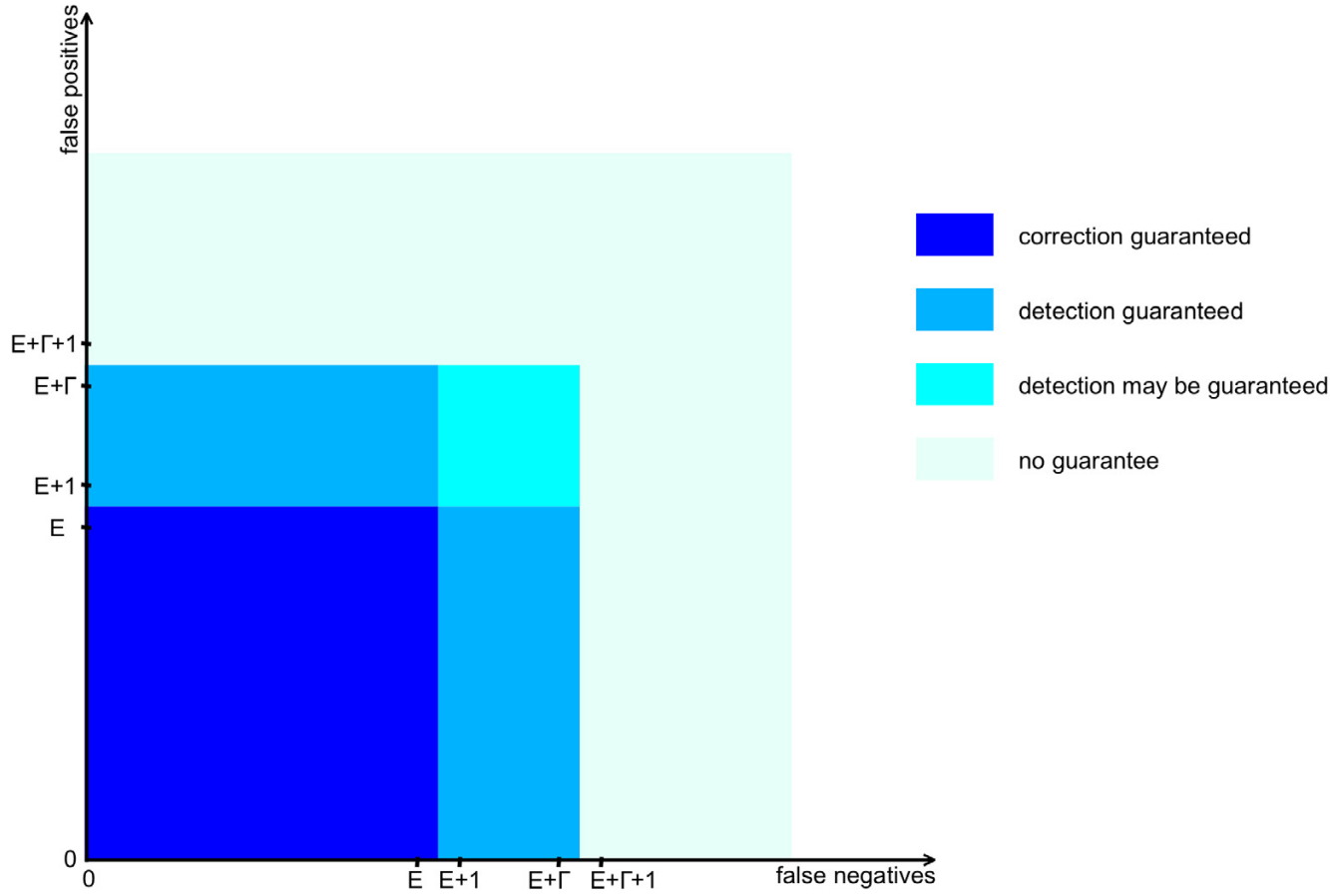

Figure 1

Guaranteed error correction and detection properties of STD. An experimenter, expecting up to t positives and E errors, chooses a satisfactory prime number q and builds the set of pools STD(n; q; t·Γ+2·E+1), as specified in corollary 2. Recall that n is the total number of variables and Γ is the compression power, i.e. the smallest γ such that $q^{\gamma+1} \geq n$. This figure summarizes the behavior of these pools when the actual number of errors exceeds E, and distinguishes between the two types of errors: false positives and false negatives. In the dark blue region, all errors are detected and corrected. In the intermediate blue rectangles, correction is not guaranteed but detection is: in an unfavorable conformation of positives and errors, correction of all errors may fail, but this failure cannot go unnoticed, and the user can therefore plan additional experiments. In the cyan square, detection is usually also guaranteed, except if E is very small (E < 2·Γ-1): in this case, the line y = 3·E+1-x splits the square in two, and detection is only guaranteed in the bottom left portion, where the total number of errors is at most 3·E+1. Finally, in the outer pale cyan zone, no guarantee is provided.

in low signal-to-noise situations, this high corrective power comes at the price of lower compression performance, since larger q values mean more pools per layer.

Corollary 2 does not distinguish between the two types of errors: false positives and false negatives. If we consider them separately, the corrective power of STD can actually be improved twofold, as shown below.

Corollary 3

Let t and E be integers such that $t \cdot \Gamma(q,n)+2 \cdot E \leq q$, and let $k = t \cdot \Gamma+2 \cdot E+1$. Consider the set of pools STD(n; q; k), and suppose that the value of each pool has been observed. Furthermore, suppose that there are at most t positive variables in $\mathcal{A}_n$, and that there are at most E false positive and E false negative observations. Then, all errors can be detected and corrected, and the value of every variable can be identified.

Proof

The proof of corollary 2 can be directly replicated, and shows that algorithm 2 still tags all variables uniquely and correctly. Indeed, since there are at most E false positives, every negative variable is tagged as such; and since there are at most E false negatives, no positive variable is tagged negative. In addition, no negative variable is tagged posi-

tive: this results from the facts that there are at most E false positives and that no positive variable is tagged negative. Finally, given that every negative variable is tagged negative, we can conclude that every positive variable is tagged positive as long as there are less than E+Γ+1 false negatives.

Error detection

If algorithm 2 tags some variables twice or not at all, or if it tags more than t variables as positive, or if it identifies more than E false positives or false negatives, then we know that the conditions for corollaries 2 and 3 are not satisfied. In this case the obtained tags may be incorrect, but one is aware of the situation. However, if enough excess errors are present, the tags can be wrong while seeming to satisfy one of the corollaries' hypotheses; in this case, the mistake is not detected. This leads to the following important question: in general, assuming there are at most t positives, how many errors can be detected?

Examining the proof of corollary 3, if there are at most E false positives and up to E+Γ false negatives, every variable is correctly tagged, although some variables may be tagged twice (i.e. both positive and negative). It follows that to avoid detection, there must be at least E+Γ+1 false negatives, or at least E+1 false positives. In fact, E+Γ+1 false negatives can successfully remain undetected. On the other hand, if there are E+1 false positives, a negative variable may seem positive with only E fictitious false negatives; but this would lead to t+1 putative positive variables, hence detection is in fact not avoided. A detailed analysis shows that escaping detection in this case actually requires either Γ extra false positives, or $2 \cdot E+1$ additional errors among which at least E+1 are false negatives. Overall, ignoring the errors' types, we conclude that the detection of $\min(3 \cdot E+1, E+\Gamma)$ errors is guaranteed. Typically Γ is 2 or 3, hence this guarantee is not very strong; but it corresponds to a rare worst case scenario, and in practice many more errors can usually be detected.

The error correction and detection properties of STD are summarized in Figure 1. From another angle, it is interesting to know what happens if more than t variables are positive. As long as there are at most E errors, all tags produced by algorithm 2 are still correct, although some variables may not be tagged (i.e., they are unresolved). Therefore the occurrence of more than t positives is detected, as in the noiseless case. However, if there are both more than E errors and more than t positives, problems may occur and escape detection (e.g., a positive variable might be "mis-tagged" as negative). Some of these problems reflect the natural limits of the STD pools, and can only be avoided by using different STD parameters; but some result from the rigidity of algorithm 2. In real applications where the number of positives and errors will probably exceed t and E in at least a few instances, more sophisticated algorithms should be used.

Even redistribution of variables

We have just shown that STD constitutes a solution to the pooling problem in the presence of experimental noise. Although it digresses from the main focus of this paper, the following theorem provides an interesting characterization of STD, basically showing that the STD layers work well together, information-wise.

Theorem 2

Let $m \le k \le q$ and consider a set of m pools $\{P_1,...,P_m\} \subset STD(n; q; k)$, each belonging to a different layer. Then:

$$\lambda_m \le \left| \bigcap_{h=1}^{m} P_h \right| \le \lambda_m + 1, \quad \text{where} \quad \lambda_m = \sum_{c=m}^{\Gamma} \left[\left\lfloor \frac{n-1}{q^c} \right\rfloor \% q \right] \cdot q^{c-m}.$$

Proof

see Methods section.

Remarks

1. λ_m depends only on m and not on the choice of $P_1,...,P_m$; hence this theorem can be expressed simply as follows: each pool is redistributed evenly in every other layer, and furthermore the intersection between any two or more pools from different layers is also redistributed evenly in the remaining layers. This property is very interesting because it means that knowing that any given pool is positive doesn't bring any information regarding which pools of another layer will be positive; hence, the information content of the other layers remains high.

2. Note that the theorem specifies $k \le q$ rather than q+1: the last layer that can be built with STD, $L(q)$, is particular and does not satisfy theorem 2.

Discussion

To evaluate and compare pooling designs, a fair performance measure is needed. A widely-used and reasonable choice consists in considering the number of pools required to guarantee the correction of all errors and the identification of all variables' values: we call this the "guarantee requirement". This criterion is used here to study the behavior and performance of STD, and to compare it to the main published deterministic error-correcting pooling designs. Since most authors do not distinguish between false positives and false negatives, we only consider here the error correction power of STD as stated in corollary 2, rather than the stronger result expressed in corollary 3.

Table 1: Choosing the optimal value for the number of pools per layer, q

q	Γ	k	v	gain
≤ 13	≥ 3	≥ 16	k > q+1, can't use these values	
17	3	16	272	36.8
19	3	16	304	32.9
23	2	11	253	39.5
29	2	11	319	31.3
...	2	11	...	...
97	2	11	1067	9.4
101	1	6	606	16.5

This table shows the gains obtained with various q values, when the total number of variables to be tested is n = 10000 and the number of expected positives is t = 5, in a noiseless experiment (E = 0). Γ is the compression power (i.e. logarithm of n in base q, see Preliminaries in Results(1) section), k is the number of layers, v is the number of pools (i.e. k·q), and the gain is defined as n/v. By construction, STD requires $k \leq q+1$; and to guarantee the identification of t positives while correcting E errors, section 3.3 showed that we must choose $k = t \cdot \Gamma + 2 \cdot E + 1$; in this example, $k = 5\Gamma + 1$. Often, the smallest useable q (i.e., satisfying $k \leq q+1$), q_{min}, yields the highest gain, but this is not always the case. In this example, $q_{min} = 17$, but q = 23 (smallest q such that Γ = 2) yields the highest gain: 39.5.

Guaranteed performance of STD

We define the "gain" of a design as the ratio between the number of variables and the number of pools: n/v. The gain is called "guaranteed gain" if the guarantee requirement is satisfied. This measure is particularly useful for comparing settings where n varies.

Given the specifications of an application, i.e. values for n (total number of objects to be tested), t (number of expected positives), and E (expected number of errors to be corrected), STD can propose many sets of pools, by selecting various values for the parameter q and setting the number of layers k accordingly (as specified by corollary 2). These pool sets are of different sizes, but all satisfy the guarantee requirement. The optimal choice, q_{opt}, is the one with maximum guaranteed gain. Let q_{min} be the smallest possible q such that $t \cdot \Gamma(q,n) + 2 \cdot E \leq q$, and let $\Gamma_{max} = \Gamma(q_{min}, n)$. At a fixed value for Γ, the number of layers k necessary to satisfy the guarantee requirement is constant; therefore the best gain at fixed Γ is always obtained with the smallest q whose compression is Γ. It follows that q_{opt} can be identified easily by finding the smallest q for each value of Γ in $\{1,...,\Gamma_{max}\}$, and calculating the corresponding gain. In practice we often have $q_{opt} = q_{min}$, but this is not compulsory, as illustrated by Table 1 in the case n = 10000, t = 5, E = 0.

The above method allows to easily calculate the best guaranteed gain that STD can offer, in any specified (n,t,E) setting. Therefore, the behavior of STD can be studied under various angles. In particular, one interesting approach consists in using fixed values for t and E, and studying the evolution of the best guaranteed gain (obtained using q_{opt}) when n increases. For example, Table 2 displays the number of pools necessary to identify three positives and correct two errors, when the number of variables ranges from 100 to 10^6. When n increases, the gain increases substantially and fairly regularly: it is multiplied by a factor ranging from 6 to 9 every time n gains an order of magnitude. Note that in a real application, the fact that the pool sizes are generally constrained by practical considerations can result in forcing to use values of $q > q_{opt}$ and hence limit the gain.

Comparison with previous work

In this section, after a brief overview of the known construction methods, we compare STD, in terms of flexibility and performance under the guarantee requirement, to the main published error-correcting deterministic pooling designs. In general, the guaranteed gains can be difficult to compare analytically, because the numbers of pools and variables can be defined by formulas that are often rather involved. However, each paper describing a new design typically holds a numerical example, which would hardly

Table 2: Gains obtained when the identification of 3 positives and the correction of 2 errors is guaranteed (t = 3, E = 2)

n	q_{opt}	pool size	k	v	gain
100	11	9	8	88	1.1
1000	11	91	11	121	8.3
10^4	13	769	14	182	55
10^5	19	5263	14	266	376
10^6	19	52631	17	323	3096

For each value of n (total number of variables), the optimal q value q_{opt} has been calculated, as well as the associated pool size, the number of layers k, the total number of pools v, and the gain.

be disadvantageous to the described design. Therefore, when the methods cannot be easily compared, it seems fair to use each paper's numerical example for comparison with STD. Note that the guarantee requirement cannot be satisfied by random designs [e.g. [8]], which are consequently not studied here.

Detailed reviews of deterministic pooling designs can be found in [7,9,10], and we will only very briefly recapitulate them here. Broadly speaking, there are three main construction methods: set packings, transversal designs, and direct constructions. In fact, the non-adaptive pooling problem is strongly connected to the problem of constructing superimposed codes [11], which was analyzed forty years ago to deal with the questions of representing rare document attributes in an information retrieval system and of assigning channels to relieve congestion in shared communications bands. The focus is different: each variable is seen as a code word and the goal is to maximize the number of code words n at fixed length v rather than the other way around; and these problems were noiseless, contrary to our own situation where error-correction is critical. Yet [11] had already suggested constructions of superimposed codes based on set packings, as well as constructions based on q-nary codes (which are in fact transversal designs) and on compositions of q-nary codes (which are not transversal anymore, and are more compact). Set packings, such as the designs presented in [12], can yield very efficient designs, but are mainly limited to $t \leq 2$ [7]. Transversal designs include the well-known grid (or row-and-column) design. This design is initially limited to identifying a single positive in the absence of noise, and is not very efficient, but it has been improved in two directions: hypercube designs [13] generalize it by considering higher dimension grids, and various methods [e.g. [14]] have been proposed to build several "synergical" grids that work well together. Finally, some authors have proposed direct constructions of error-correcting pooling designs [15,16].

Note that STD, although directly constructed, is in fact a transversal design. Furthermore, STD can be seen as a constructive definition of a q-nary code as proposed by [11], i.e. a concatenated code where the inner code is simply the unary code, and the outer code has some similarities with a Reed-Solomon code [17]. Yet although related, the methods are clearly different: for example, STD doesn't produce useful pools if q is a prime power; on the other hand, STD allows to build up to q+1 layers, whereas the Reed-Solomon based construction can only build up to q-1. Furthermore, STD produces efficient pools independently of the number of variables n, contrary to the Reed-Solomon approach where one is faced with the difficult problem of choosing a good subset of code words except for some n values. The relationship between the two approaches requires further investigations.

Set packing designs

Regarding set packing designs, the main results taking into account error-correction are presented in [12]. The authors exhibit Steiner designs that can identify up to t = 2 positives and in some instances correct many errors, and prove that these designs are optimal when the construction is possible (it is only possible for very specific (n,E) values). When these optimal designs exist, they are more efficient than STD. The same authors describe a real-world application in [18], where the goal is to screen a clone map with n = 1530 and t = 2. They start off with a design that can deal with 4368 variables while satisfying the guarantee requirement for t = 2 and E = 0. None of the optimal designs from [12] can be used, but this initial design is also based on a Steiner system and remains very efficient. The authors then select 1530 of the 4368 variables to serve as clones in their experiment. This was presumably done because Steiner systems, even outside the optimality conditions of [12], are not known for arbitrary values of n. Although this reduces the resulting designs' performance, they remain efficient and obviously still satisfy the guarantee requirement. Additionally, this strategy reduces the sizes of pools, providing increased robustness (e.g., some information can still be obtained if, exceptionally, three objects are positive), and complying with the application-imposed pool size constraints. In the example, n = 1530 and t = 2, and the authors propose two designs: one with 65 pools of approximately 118 clones each, and one with 54 pools of 142 clones. These numbers are very close to what would be recommended with STD: we could propose STD(1530; 13; 5) which has 65 pools of 118 clones, or STD(1530; 7; 7) with 49 pools of 218 clones. Note that although STD(1530; 13; 5) has the same number of pools and pool size as the first design proposed in [18], they are in fact different: the latter is obtained by random sampling from the Steiner design. All of these designs guarantee the identification of 2 positives in the absence of noise. Furthermore, although noise-tolerance is not guaranteed in any of them, simulations we have performed suggest that substantial error-rates can be corrected in the STD designs, as is the case in the others. Therefore these designs and STD appear to achieve very similar performances on these examples. However, it is important to note that the only Steiner systems proven to be optimal concern specific instances of the t = 1 and t = 2 cases. In more general circumstances, designs derived from Steiner systems are not optimal, and their performance depends on the problem specification (i.e. n, t, E values). For example, considering the n = 10000, t = 5, E = 0 problem discussed above and in Table 1, the smallest Steiner system that we could identify (based on [19]) is S(3,24,530), which comprises 530 pools. In addition, there is no clear method for

choosing the Steiner system best suited to a given problem specification: although we have searched extensively, we cannot be certain that no better Steiner system exists for this example. In contrast, finding the optimal STD parameters is straight-forward, as explained in the previous section. In this case STD proposes a solution comprising 253 pools.

Transversal designs

An interesting generalization of the grid design is described in [13]. The authors propose to array the variables in a D-dimensional cube, instead of the 2 dimensions used in the standard grid design. Furthermore, they advise that the length of the cube's side be chosen prime: let us denote it q. A pool is then obtained from each hyperplane, so that the D-dimensional cube yields D layers of q pools, each comprising up to n/q variables. To obtain more layers, the authors propose a criterion to construct "efficient transforming matrices" that produce additional cubes, where variables are as shuffled as possible; in fact, the purpose of their "efficiency" criterion is identical to the "co-occurrence of variables" property satisfied by STD (theorem 1). Seen like this, their system is clearly related to STD: D is Γ+1, and although the authors do not investigate their design's behavior under the guarantee requirement, corollaries 1 and 2 can in essence be applied. Furthermore, when the cube is "full", i.e. when $n = q^D$, their pools satisfy an analog of theorem 2 (i.e. they are "information-efficient" in some sense). Note that this cannot be the case when q is arbitrary; this may explain why the authors limit their options for q to the smallest primes larger than $n^{1/D}$, for each D value. However, each D-dimensional cube provides only D layers, and the proposed criterion for building additional cubes is not systematic, so that the total number of layers that can be built is unclear but seems much smaller than with STD. In addition, the authors don't take observational noise into account (they do talk of "false positives", but are really referring to what we call ambiguous variables). For these reasons, we cannot rigorously compare the designs under the guarantee requirement, but in general the fact that STD can build more layers is clearly favorable, since it allows dealing with a greater number of positives and/or errors at any chosen q value. In a numerical example concerning the screening of the CEPH YAC library, n = 72000 and the authors argue that the optimal dimension and side length to use are D = 3 and q = 43, respectively. They then exhibit a set of transforming matrices that allows the construction of at most 3 additional cubes, yielding a total of 12 layers. By contrast, using the same values for D and q, STD can build up to 44 layers, which all satisfy the efficiency criterion. We believe that some of these extra layers could prove valuable, especially when allowing for experimental noise. In addition, smaller values for q can be used with STD (while still being information-efficient in the sense of theorem 2), although simulations would be necessary to choose the best value.

Two other transversal pooling designs, which generalize the grid design by providing additional 2-dimensional grids, are described in [14]: the "Union Jack" and the RCF designs. In essence, they are very similar to STD when Γ = 1: writing $q = \sqrt{n}$, they allow the construction of up to q+1 layers of pools (where each layer contains q pools of size q) which satisfy the property that any pair of variables appears in at most one pool. Theorem 1 shows that this property, known as the "unique colinearity" condition, is in fact verified by STD when Γ = 1 (in accord with $q = \sqrt{n}$). We can observe that these designs, as well as STD when Γ = 1, are maximal under this condition, since each pair of variables is in fact present in exactly one pool. Corollaries 1 and 2 can be applied, and show that they allow the identification of up to t positives while correcting E observation errors, provided that $t+2 \cdot E+1 \leq q+1$. The performance of the designs from [14] is therefore identical to that of STD when Γ = 1. However, STD is superior to these designs in two respects. First, their constructions are only possible if q is prime and $q \equiv 5 \bmod 6$ (using the RCF construction), or if q is prime and $q \equiv 3 \bmod 4$ (with the Union Jack design). By contrast, STD only requires that q is prime. Second, STD can be used with any compression power, rather than being limited to Γ = 1. This flexibility is an advantage, because STD can be customized to suit more applications. Notably, when the fraction of positives is small, the Union Jack and RCF designs perform less well: the pools are too small, and observing that a pool is negative brings little information. By contrast, pools in STD can be very large (when choosing a small q), so that every observation is informative. To illustrate this point, let us consider the numerical example of [16] discussed below, where the fraction of positives is particularly low (n = 18,918,900 and t is 2 or 9). The best usable design from [14] would be a Union Jack with q = 4363, and would require a total of 13,089 pools for 2 positives – 77 times more than STD – and 43,630 pools to guarantee the identification of 9 positives – 32 times more than STD.

Direct constructions

In [15], the author proposes a direct construction allowing the detection of an arbitrary number of positives. Although this design is not very efficient under the guarantee requirement, the author shows in [20] that the pools designed for detecting 2 positives allow with high probability the detection of more positives. A numerical example, presented in [9], is the following. If $n = 10^6$ and t = 5, using 946 pools guarantees the identification of 2 positives and successfully identifies up to 5 positives with probability 97.1%. In comparison, under the guarantee requirement (i.e. with probability 100%), STD(n; 11; 11) contains 121 pools and identifies 2 positives, and STD(n;

23; 21), which comprises 483 pools, guarantees the identification of up to 5 positives.

Finally, another group [16] described two new classes of non-adaptive pooling designs, which allow the detection of any number of positives and the correction of half as many errors. Following the idea from [20], they also show that their designs for t = 2 have high probabilities of being successful for more positives. In a numerical example, they consider the case n = 18,918,900, and propose a design with 5460 pools which guarantees the identification of 2 positives, and can in addition identify up to 9 positives with 98.5% chance of success. By contrast, STD(n; 13; 13) contains 169 pools and guarantees the identification of 2 positives, and the identification of 9 positives is guaranteed with the 1369 pools of STD(n; 37; 37).

Conclusion

In this paper, we have presented a new pooling design: the "shifted transversal design" (STD). We have proven that it constitutes an error-correcting solution to the pooling problem. This design is highly flexible: it can be tailored to deal efficiently with many experimental settings (i.e., numbers of variables, positives and errors). Finally, under a standard performance criterion, i.e. requiring that the correction of all errors and the identification of all variables' values be guaranteed mathematically, we have shown that STD compares favorably, in terms of numbers of pools, to the main previously described deterministic pooling designs.

This approach is being experimentally validated in collaboration with Marc Vidal's laboratory at the Dana Farber Cancer Institute, Boston. In a pilot project, pools have been generated with 940 AD-Y preys, using the STD(940;13;13) design, and we are screening the 169 resulting pools against 50 different baits. This experiment will provide estimations for the technical noise levels of their high-throughput 2-hybrid protocol, in addition to producing valuable interaction data and yielding a real-world evaluation of STD.

Although this work is motivated by protein interaction mapping, as we have been collaborating with Marc Vidal's group for several years, its scope is certainly not limited to high-throughput two-hybrid projects. Potential applications include a wide range of high-throughput PCR-based assays such as gene knockout projects, drug screening projects, and various proteomics studies. Furthermore, this general problem certainly has applications outside biology.

In practice, an important point is made in [20], where the author shows that his pooling design can be used to detect with high probability more positives than guaranteed. Simulations we have performed show that this observation is also true with STD: the gains can be increased substantially if one tolerates a small fraction of ambiguous variables that will need to be retested. However, these considerations are outside the scope of this paper, because we cannot study them analytically, but resort instead to computer simulations. Yet using such a strategy in practice with STD significantly improves the performance. For example, consider the case n = 10000 and t = 5, and suppose that the assay has an error-rate of 1%. To guarantee the identification of all variables' values, one must use 483 pools (with q = 23 and k = 21). However, if one tolerates up to 10 ambiguous variables, even when overestimating the error-rate to 2% for safety's sake, 143 pools prove amply sufficient. It is clear that this "ambiguity-tolerant" approach should be preferred in practical applications. This approach and the corresponding computer program, which performs simulations to select the STD parameter values best suited for a given application and includes original efficient algorithms for preparing the pools and decoding the outcomes, will be discussed in another paper.

Another interesting track will be to study the efficiency of pooling designs from the point of view of Shannon's information theory. We are planning to investigate STD's behavior in this context. Theorem 2 could prove useful for this.

Finally, the connection between STD and constructions based on superimposed codes, e.g. q-nary Reed-Solomon codes [11], warrants further studies.

Methods

Proof of theorem 1

Let $i_1, i_2 \in \{0,...,n-1\}$ with $i_1 \neq i_2$. Since each layer of pools is a partition of $\mathcal{A}_n$, there cannot be more than one pool per layer containing both A_{i1} and A_{i2}. Furthermore, there exists a pool in layer $L(j)$ that contains both A_{i1} and A_{i2} if and only if the columns for A_{i1} and A_{i2} are equal in M_j, that is to say $C_{j,i_1} = C_{j,i_2}$. Therefore the number of pools of STD(n; q; q+1) that contain both i_1 and i_2, $\mathrm{Card}(pools_{q+1}(i_1) \cap pools_{q+1}(i_2))$, is the number of values of j in {0,...,q} such that $C_{j,i_1} = C_{j,i_2}$. However, the following equivalencies hold $\forall\ j \in \{0,...,q-1\}$:

$$C_{j,i_1} = C_{j,i_2} \quad \Leftrightarrow \quad s(i_1, j) \equiv s(i_2, j) \mod q$$
$$\Leftrightarrow \sum_{c=0}^{\Gamma} j^c \cdot \left\lfloor \frac{i_1}{q^c} \right\rfloor \equiv \sum_{c=0}^{\Gamma} j^c \cdot \left\lfloor \frac{i_2}{q^c} \right\rfloor \mod q$$
$$\Leftrightarrow \sum_{c=0}^{\Gamma} j^c \cdot \left(\left\lfloor \frac{i_1}{q^c} \right\rfloor - \left\lfloor \frac{i_2}{q^c} \right\rfloor \right) \equiv 0 \mod q \qquad (1)$$

Since q is prime, Z/qZ is a field, namely the Galois field $GF(q)$.

Furthermore, since $i_1 \neq i_2$, there exists at least one value $c \in \{0,...,\Gamma\}$ such that $\left(\left\lfloor \frac{i_1}{q^c} \right\rfloor - \left\lfloor \frac{i_2}{q^c} \right\rfloor \right) \neq 0 \mod q$. Indeed, $i_1, i_2 \in \{0,...,n-1\}$ and $n \leq q^{\Gamma+1}$ entails that $i_1 = \sum_{c=0}^{\Gamma} \left(\left\lfloor \frac{i_1}{q^c} \right\rfloor \% q \right) \cdot q^c$ and $i_2 = \sum_{c=0}^{\Gamma} \left(\left\lfloor \frac{i_2}{q^c} \right\rfloor \% q \right) \cdot q^c$, where % denotes the modulus (these are the unique decompositions of i_1 and i_2 in base q). Hence, $i_1 - i_2 = \sum_{c=0}^{\Gamma} \left(\left(\left\lfloor \frac{i_1}{q^c} \right\rfloor - \left\lfloor \frac{i_2}{q^c} \right\rfloor \right) \% q \right) \cdot q^c$. Supposing that $\left(\left\lfloor \frac{i_1}{q^c} \right\rfloor - \left\lfloor \frac{i_2}{q^c} \right\rfloor \right) \equiv 0 \mod q$ for every $c \in \{0,...,\Gamma\}$ would lead to $i_1 - i_2 = 0$, which is contradictory with the hypothesis that $i_1 \neq i_2$.

It follows that the above (1) can be seen as a non-zero polynomial (in j) of degree at most Γ on $GF(q)$. As is well-known, such a polynomial has at most Γ roots in $GF(q)$. That is to say, there are at most Γ values of j in {0,...,q-1} such that a pool of $L(j)$ contains both A_{i1} and A_{i2}. This proves the theorem if $C_{q,i_1} \neq C_{q,i_2}$. Furthermore, if $C_{q,i_1} = C_{q,i_2}$, the coefficient of j^Γ in (1) is zero by definition of $s(i,q)$, and (1) is of degree at most Γ-1. Therefore if A_{i1} and A_{i2} are elements of the same pool in $L(q)$, then there are at most Γ-1 pools in $L(0),...,L(q-1)$ that contain both A_{i1} and A_{i2}. This concludes the proof of theorem 1.

Proof of theorem 2

Let $j_1,...,j_m \in \{0,...,k-1\}$ be the layer numbers and $p_1,...,p_m \in \{0,...,q-1\}$ be the pool indexes that define $\{P_1,...,P_m\}$: for every $h \in \{1,...,m\}$, P_h contains all variables of index i $\in \{0,...,n-1\}$ such that $s(i,j_h) \equiv p_h \mod q$.

$\left| \bigcap_{h=1}^{m} P_h \right|$ is the number of values $i \in \{0,...,n-1\}$ such that: $\forall h \in \{1,...,m\}$,

$s(i,j_h) \equiv p_h \mod q$. Writing $i = \sum_{c=0}^{\Gamma} \alpha_c \cdot q^c$ with $\alpha_0,...,\alpha_\Gamma \in \{0,...,q-1\}$ (this is the unique decomposition of i in base q), the above is equivalent to:

$$\forall h \in \{1,\ldots,m\}, \sum_{c=0}^{\Gamma} \alpha_c \cdot j_h^c \equiv p_h \mod_q . \qquad (2)$$

This system can be written:

$$\begin{bmatrix} 1 & j_1 & j_1^2 & \cdots & j_1^\Gamma \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & j_m & j_m^2 & \cdots & j_m^\Gamma \end{bmatrix} \cdot \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_\Gamma \end{bmatrix} \equiv \begin{bmatrix} p_1 \\ \vdots \\ p_m \end{bmatrix} \mod q.$$

If m ≥ Γ+1: consider the square sub-matrix composed of the first Γ+1 rows of the left member. Since $P_1,...,P_m$ belong to different layers, the j_h values are all distinct. Therefore, recalling that q is prime, this sub-matrix can be seen as a Vandermonde matrix with elements in the Galois field $GF(q)$: it is nonsingular. This shows the existence of a unique tuple of values for $\alpha_0,...,\alpha_\Gamma \in \{0,...,q-1\}$ satisfying the first Γ+1 congruencies of (2). The remaining m-(Γ+1) congruencies may or may not be satisfied with these $\alpha_0,...,\alpha_\Gamma$ values, and the corresponding $i = \sum_{c=0}^{\Gamma} \alpha_c \cdot q^c$ might be too large (i.e. ≥ n); but in any case, there is at most one value of i satisfying the system: theorem 2 is proved when m ≥ Γ+1 (given that in this case $\lambda_m = 0$).

Otherwise, m ≤ Γ: consider the square sub-matrix composed of the first m columns of the left member. Again, this sub-matrix is a Vandermonde matrix in $GF(q)$, hence it is nonsingular. Consequently, given any values for $\alpha_m,...,\alpha_\Gamma$, there exists a unique tuple of values for $\alpha_0,...,\alpha_{m-1}$ in {0,...,q-1} satisfying (2) (simply shift the terms in $\alpha_m,...,\alpha_\Gamma$ to the right member). The question therefore becomes: how many tuples of values for $\alpha_m,...,\alpha_\Gamma$ exist, such that $i = \sum_{c=0}^{\Gamma} \alpha_c \cdot q^c < n$ where $\alpha_0,...,\alpha_{m-1}$ are determined by $\alpha_m,...,\alpha_\Gamma$ as explained above. To answer this,

consider the unique decomposition of n-1 in base q: $n-1=\sum_{c=0}^{\Gamma}\beta_c\cdot q^c$, where $\beta_c=\left\lfloor\frac{n-1}{q^c}\right\rfloor\%q$ for $c\in\{0,...,\Gamma\}$. Under this representation, it is clear that $i < n$, i.e. $i \le n-1$, if and only if:

$\alpha_\Gamma < \beta_\Gamma$ or

$$(\alpha_\Gamma=\beta_\Gamma \text{ and } (\alpha_\Gamma<\beta_\Gamma \text{ or } (\ldots \text{ and } (\alpha_m<\beta_m \text{ or } (\alpha_m=\beta_m \text{ and } \sum_{c=0}^{\Gamma} a_c\cdot q^c < n))\ldots))).$$

For each $c\in\{m,...,\Gamma\}$, the branch ending at $(\alpha_c<\beta_c)$ yields $\beta_c\cdot q^{(c-m)}$ different tuples. Indeed, for $d > c$ $\alpha_d=\beta_d$ in this branch, and $\alpha_0,...,\alpha_{m-1}$ are bound to $\alpha_m,...,\alpha_\Gamma$: there are β_c possible choices for α_c, and q choices each for $\alpha_m,...,\alpha_{c-1}$. As to the final branch, it can yield at most one solution, given that all the α values are set or bound in this branch.

Consequently, there are a total of $\lambda_m=\sum_{c=m}^{\Gamma}\beta_c\cdot q^{c-m}$ or λ_m+1 solutions: theorem 2 is also proved when $m \le \Gamma$.

Acknowledgements

I thank Danielle Thierry-Mieg, Jean Thierry-Mieg, Laurent Trilling and Jean-Louis Roch for stimulating discussions and for carefully reading the manuscript, and an anonymous reviewer for insightful comments. This work was funded by a BQR grant from the Institut National Polytechnique de Grenoble (INPG) to NT.

References

1. Reboul J, Vaglio P, Rual JF, Lamesch P, Martinez M, Armstrong CM, Li S, Jacotot L, Bertin N, Janky R, Moore T, Hudson JR Jr, Hartley JL, Brasch MA, Vandenhaute J, Boulton S, Endress GA, Jenna S, Chevet E, Papasotiropoulos V, Tolias PP, Ptacek J, Snyder M, Huang R, Chance MR, Lee H, Doucette-Stamm L, Hill DE, Vidal M: ***C. elegans* ORFeome version 1.1: experimental verification of the genome annotation and resource for proteome-scale protein expression.** *Nat Genet* 2003, **34(1):**35-41.
2. Walhout A, Sordella R, Lu X, Hartley J, Temple GF, Brasch MA, Thierry-Mieg N, Vidal M: **Protein interaction mapping in *C. elegans* using proteins involved in vulval development.** *Science* 2000, **287:**116-122.
3. Davy A, Bello P, Thierry-Mieg N, Vaglio P, Hitti J, Doucette-Stamm L, Thierry-Mieg D, Reboul J, Boulton S, Walhout AJ, Coux O, Vidal M: **A protein-protein interaction map of the *Caenorhabditis elegans* 26S proteasome.** *EMBO Rep* 2001, **2(9):**821-828.
4. Evans G, Lewis K: **Physical mapping of complex genomes by cosmid multiplex analysis.** *Proc Natl Acad Sci USA* 1989, **86(13):**5030-5034.
5. Zwaal R, Broeks A, van Meurs J, Groenen J, Plasterk RH: **Target-selected gene inactivation in *Caenorhabditis elegans* by using a frozen transposon insertion mutant bank.** *Proc Natl Acad Sci USA* 1993, **90(16):**7431-7435.
6. Cai W, Chen R, Gibbs R, Bradley A: **A clone-array pooled shotgun strategy for sequencing large genomes.** *Genome Res* 2001, **11(10):**1619-1623.
7. Balding D, Bruno W, Knill E, Torney D: **A comparative survey of non-adaptive pooling designs.** In *Genetic mapping and DNA sequencing* New York: Springer; 1996:133-154.
8. Bruno W, Knill E, Balding D, Bruce D, Doggett NA, Sawhill WW, Stallings RL, Whittaker CC, Torney DC: **Efficient pooling designs for library screening.** *Genomics* 1995, **26:**21-30.
9. Ngo H, Du DZ: **A survey on combinatorial group testing algorithms with applications to DNA library screening.** *DIMACS Ser Discrete Math Theoret Comput Sci* 2000, **55:**171-182.
10. Du DZ, Hwang F: **Combinatorial Group Testing and Its Applications, 2nd edn.** Singapore: World Scientific; 2000.
11. Kautz W, Singleton H: **Nonrandom binary superimposed codes.** *IEEE Trans Inform Theory* 1964, **10:**363-377.
12. Balding D, Torney D: **Optimal pooling designs with error correction.** *J Comb Theory Ser A* 1996, **74:**131-140.
13. Barillot E, Lacroix B, Cohen D: **Theoretical analysis of library screening using a N-dimensional pooling strategy.** *Nucl Acids Res* 1991, **19:**6241-6247.
14. Chateauneuf M, Colbourn C, Kreher D, Lamken E, Torney D: **Pooling, lattice square, and union jack designs.** *Ann Comb* 1999, **3:**27-35.
15. Macula A: **A simple construction of d-disjunct matrices with certain constant weights.** *Discrete Math* 1996, **162(1–3):**311-312.
16. Ngo H, Du DZ: **New constructions of non-adaptive and error-tolerance pooling designs.** *Discrete Math* 2002, **243(1–3):**161-170.
17. Reed I, Solomon G: **Polynomial codes over certain finite fields.** *J Soc Ind Appl Math* 1960, **8:**300-304.
18. Balding D, Torney D: **The design of pooling experiments for screening a clone map.** *Fungal genet, biol* 1997, **21:**302-307.
19. Colbourn C, Mathon R: **Steiner systems.** In *The CRC Handbook of Combinatorial Designs* Edited by: Colbourn C, Dinitz J. Boca Raton: CRC Press; 1996:66-75.
20. Macula A: **Probabilistic nonadaptive group testing in the presence of errors and dna library screening.** *Ann Comb* 1999, **3:**61-69.

BIOINFORMATICS ORIGINAL PAPER
Vol. 24 no. 5 2008, pages 696–703
doi:10.1093/bioinformatics/btn001

Systems biology

Interpool: interpreting smart-pooling results

Nicolas Thierry-Mieg* and Gilles Bailly†

TIMC-IMAG, CNRS UMR5525, Faculte de Medecine, 38706 La Tronche Cedex, France

Received on July 6, 2007; revised on October 17, 2007; accepted on January 1, 2008

Advance Access publication January 9, 2008

Associate Editor: Alfonso Valencia

ABSTRACT

Motivation: In high-throughput projects aiming to identify rare positives using a binary assay, smart-pooling constitutes an appealing strategy liable of significantly reducing the number of tests while correcting for experimental noise. In order to perform simulations for choosing an appropriate set of pools, and later to interpret the experimental results, the pool outcomes must be 'decoded'. The intuitive aim is clearly to identify the positives that gave rise to an observation, whether real or simulated. However, this goal is not well-formalized and has been the focus of very few studies.

Results: We first provide a clear combinatorial formalization of the 'decoding problem'. We then present *interpool*, an exact algorithm to solve this problem. An efficient implementation is freely available. Its usefulness is illustrated in the context of yeast-two-hybrid interactome mapping with the Shifted Transversal Design.

Availability: The implementation, licensed under the GNU GPL, can be downloaded from http://www-timc.imag.fr/Nicolas.Thierry-Mieg/

Contact: nicolas.thierry-mieg@imag.fr

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

Many high-throughput projects rely on a basic yes-or-no test, and aim to identify a few rare positives in a large collection of 'objects' (clones, proteins, peptides, etc.). The main issue is obviously to obtain information as efficiently as possible, but another major difficulty stems from the fact that biological assays can be somewhat noisy: false positives and false negatives can and do occasionally occur. However, often the basic assay can be applied to a pool of objects, yielding a positive readout if the pool contains at least one positive. When this is the case, smart-pooling constitutes an appealing approach that can significantly reduce the number of tests while providing the power to correct the experimental errors (Barillot *et al*., 1991; Bruno *et al*., 1995; Jin *et al*., 2007; Thierry-Mieg, 2006a; Vermeirssen *et al*., 2007).

The strategy consists in assaying well-chosen pools, such that each object is present in several pools (i.e. the pools are redundant). Pools are designed so that all positive objects can usually be identified from the pattern of positive pools, and when this is not the case, only a few candidates need to be retested. In addition, the pools' redundancy means that each object is tested several times: this provides a potential increase in both sensitivity and specificity.

The first difficulty is to choose a 'good' set of pools. This is the focus of the so-called pooling problem, or combinatorial group testing problem: given bounds on the numbers of positives and errors, this problem consists in designing a set of pools that guarantees the correction of all errors and the identification of all positives. Several mathematical constructions satisfying this so-called 'guarantee requirement' have been described (e.g. Thierry-Mieg, 2006b and references therein).

Once the design has been chosen and the pools are constructed, they can be assayed in every 'condition' of interest, giving rise to an observation for each condition. For example in a yeast-two-hybrid experiment, the preys can be smart-pooled and then screened against individual baits, as in Jin *et al*. (2007). Each bait can be seen as a condition, and the resulting observation takes the form of a score for each pool (often simply 'negative' or 'positive'). The goal is then to identify the positive objects (preys) that produced this observation. Whether this is achievable or impossible depends on the pooling design, and also on the numbers of positives and errors that actually occurred. But even when it is theoretically feasible, doing it is not algorithmically trivial: this is referred to as the decoding problem, although it has never been formally defined as far as we know.

Proving that a pooling design satisfies the guarantee requirement, is sometimes achieved by exhibiting a provably correct decoding algorithm (e.g. Thierry-Mieg, 2006b). However, such algorithms rely on the given bounds on numbers of positives and errors. In real experiments, these bounds are unknown but can be very large. Attempting to estimate and use the true bounds when selecting the set of pools would lead to inefficient designs, with many pools that would be useless except perhaps in extremely rare instances. In practice, if one accepts that in a small fraction of instances the experiment will fail to identify every positive, designs with many fewer pools can be used. A drawback is that the previous decoding algorithms cannot be used: more general algorithms that do not assume knowledge of the maximum numbers of positives and errors are needed.

Such general algorithms are also crucial for selecting a pooling design well-adapted to the experimental context. Indeed, the numbers of positives and errors will vary in each tested condition, and can usually only be roughly estimated beforehand. Large numbers of simulations should be performed, using

*To whom correspondence should be addressed.
†Present address: LIG, University of Grenoble 1, BP 253, 38041 Grenoble Cedex 9, France.

various reasonable values for the expected rates of positives and errors, in order to find the right compromise between minimizing the number of pools that have to be built and screened, and maximizing the sensitivity and specificity in the conditions of interest. Consequently, the algorithms must be efficient.

A decoding algorithm called the Markov Chain Pool Decoder (MCPD) has been proposed (Knill *et al*., 1996), and an implementation is available. Given an observation, it estimates the posterior probability that each object is positive. The formal problem addressed by MCPD is related to the decoding problem implicit in combinatorial group testing and to its generalization presented in Section 2: there is a link between the likelihood function of (Knill *et al*., 1996) and the distance minimized in the combinatorial problem, although determining the exact relationship would require further studies. However, because it relies on a Markov Chain Monte Carlo method, MCPD's accuracy depends on the number of steps and there is no easy way to know when convergence has been attained. In addition, although it is fast enough to be used for decoding real observations, its speed becomes a limiting factor when performing large numbers of simulations. Finally, MCPD is a stochastic algorithm, while we wished to study the decoding problem from the deterministic point of view, maintaining a direct link with the classic decoding problem implicit in combinatorial group testing.

In this article, we present a new exact algorithm for interpreting smart-pooling results: *interpool*. We first provide a clear formalization of the decoding problem. In Section 3, we give the theoretical basis of our method. We finally present the algorithm, and illustrate its use in the context of an interactome mapping project.

2 FORMALIZING THE DECODING PROBLEM

NOTATION. *Let $\mathcal{V}$ be a set of Boolean variables. We call pool a subset of $\mathcal{V}$. A pooling design, noted $\mathcal{D}$, is a set of pools. In the whole article, we will consider that $\mathcal{V}$ and $\mathcal{D}$ are given once and for all.*

When performing an experiment, each pool produces a signal which is a priori continuous, although it should hopefully be highly contrasted for high-thoughput yes-or-no assays. This signal is then interpreted by the user or image-analysis software to obtain a discrete outcome for each pool, typically chosen among the two values 'positive' and 'negative'. However, in general it could be interesting to use more than two values, because higher confidence can be placed into the strongest and weakest signals than in the intermediate ones. In this context, assaying a set of pools in a given condition yields an observation, defined as follows.

NOTATION. *Let Ω be the set of values representing the possible discrete outcomes of a pool.*

For example, one could allow four outcomes of increasing strength for each pool with $\Omega = \{\text{NONE}, \text{FAINT}, \text{WEAK}, \text{STRONG}\}$.

DEFINITION 1. *An observation is a mapping between $\mathcal{D}$ and Ω: to each pool, it associates its discretized outcome. Equivalently, an observation can be seen as a vector in $\Omega^{|\mathcal{D}|}$.*

Given an observation, the goal is to identify the positive variables. An implicit hypothesis is that although the assays may yield more or less continuous signals, the variables themselves are truly Boolean. For example, in the context of yeast-two-hybrid—and particularly in a high-throughput setting—we must assume that either a pair of proteins do have the potential to interact physically, or they do not. It follows that conceptually, each pool's outcome should ideally also be Boolean: in the absence of noise, a pool should be positive if and only if it contains a positive variable. This leads to the following definitions.

DEFINITION 2. *An interpretation is a mapping between $\mathcal{D}$ and $\{0, 1\}$, or equivalently it is a vector in $\{0, 1\}^{|\mathcal{D}|}$: to each pool, it associates a unique value in $\{0, 1\}$ meaning that the pool is respectively negative or positive in this interpretation.*

DEFINITION 3. *An interpretation $\imath$ is consistent if there exists a mapping between $\mathcal{V}$ and $\{0, 1\}$ such that the value of each pool, defined as the disjunction (logical OR) of the pool's variables' values in this mapping, is equal to its value in the interpretation $\imath$.*

NOTATION. *Let $\mathcal{C}$ be the set of consistent interpretations.*

$\mathcal{C}$ is totally determined by $\mathcal{V}$ and $\mathcal{D}$, although it is typically huge. For example, in a modest interactome project where we wish to detect up to five positives among 1000 preys, there must be at least $\binom{1000}{5} \approx 2^{50}$ different consistent interpretations. Hence $\mathcal{C}$ cannot be computed. However, the following simple algorithm allows to test whether an interpretation is consistent, and simultaneously decode it if it is:

(1) variables that appear in a negative pool are negative;
(2) the remaining variables are positive if they are the only non-negative variables in at least one positive pool, and 'ambiguous' otherwise.
(3) Finally, if every positive pool contains at least one non-negative variable, the interpretation is consistent.

Ambiguous variables are those for which there is no clear evidence one way or the other. These need to be retested in confirmatory assays if completeness is sought. However, with a well-chosen pooling design, they should only occur when the number of positives is much larger than expected.

In order to decode an observation, the only component that is still missing is a relationship between observations and interpretations. First, a mapping between Ω and $\{0, 1\}$ has to be specified: one must decide whether each outcome of Ω is to be preferentially interpreted as positive or negative. When $|\Omega| = 2$, there are only two possible outcomes: we can choose to call them NEG and POS, i.e. $\Omega = \{\text{NEG}, \text{POS}\}$, which can be naturally mapped to $\{0, 1\}$. In the general case where $|\Omega| > 2$, it is less trivial yet the mapping is usually implicit to the choice of the values and cutoffs (with regards to the underlying continuous signal) that define Ω.

NOTATION. *We will note $\mathcal{N}$ and $\mathcal{P}$ the subsets of Ω that map to 0 and 1, respectively.*

Given such a mapping between Ω and $\{0, 1\}$, any observation can be associated to the unique interpretation induced by

this mapping. This is called the observation's canonical interpretation. In the absence of experimental errors, the canonical interpretation of any observation is necessarily consistent. Indeed, the mapping that associates each variable to its true value clearly works. But in general, an observation's canonical interpretation can be inconsistent. In fact, if $\mathcal{D}$ is well-chosen, this should always be the case when the observation contains errors. Therefore, we need to specify the relationship between an observation and an arbitrary interpretation.

To this end, let us assume given a 'distance' δ between each element of Ω and each of 0 and 1. This distance $\delta : \Omega \times \{0,1\} \to I\!N$ must satisfy the following rules:

- the distance between $\omega \in \Omega$ and its canonical mapping is 0;
- the distance between $\omega \in \Omega$ and the element of $\{0,1\}$ that it does not map to, noted δ_ω, is a strictly positive integer.

By extension, this also defines the distance between an observation and an interpretation, simply by summing the distances over all pools as follows.

DEFINITION 4. *The distance between an observation* o *and an interpretation* ı *is defined as:*

$$\delta(o, \imath) \triangleq \sum_{p \in \mathcal{D}} \delta(o(p), \imath(p)) .$$

Note that δ is fully defined by $\mathcal{N}$, $\mathcal{P}$ and the value of δ_ω for each $\omega \in \Omega$.

DEFINITION 5. *We can now formally define the decoding problem as we see it. Consider given a set of possible assay outcomes* Ω *and a distance* δ. *For any observation* o, *the goal is to identify the set* $\mathcal{S}(o)$ *of consistent interpretations at minimal distance to* o: $\mathcal{S}(o) \triangleq \{\imath \in \mathcal{C} \mid \delta(o, \imath) = \Delta(o)\}$, *where* $\Delta(o) \triangleq min_{\imath \in \mathcal{C}}\{\delta(o, \imath)\}$, *i.e.* $\Delta(o)$ *is the distance between* o *and its nearest consistent interpretation*(*s*).

This general framework is illustrated in the following three increasingly flexible instances.

First, consider the case $\Omega = \{\text{NEG}, \text{POS}\}$. We can define δ by $\delta(\text{NEG}, 0) = \delta(\text{POS}, 1) = 0$, and $\delta(\text{NEG}, 1) = \delta(\text{POS}, 0) = 1$. Equivalently and perhaps more intuitively, this same distance could be defined by $\mathcal{N} = \{\text{NEG}\}$, $\mathcal{P} = \{\text{POS}\}$, and $\delta_{\text{NEG}} = \delta_{\text{POS}} = 1$ [recalling that in this case $\delta_{\text{NEG}} = \delta(\text{NEG}, 1)$ and $\delta_{\text{POS}} = \delta(\text{POS}, 0)$]. The distance between an observation and an interpretation is then the standard Hamming distance. This first instance of the decoding problem is precisely the one that is implicit in the context of the combinatorial group testing problem, where the error model is simply defined by the total number of errors. Note that although in this case we have a true metric distance in the topological sense, this is not required in our framework, and is not true in the following instances.

A limitation of the above is that it makes no distinction between false positives and false negatives. However, in many assays the error-rates can be very different for these two types of errors. This can be taken into account, still using $\Omega = \{\text{NEG}, \text{POS}\}$, by defining the distance δ such that $\delta_{\text{NEG}} \neq \delta_{\text{POS}} : \delta_{\text{NEG}}$ and δ_{POS} can be seen as the respective 'costs' of false negative and false positive outcomes. Different choices for δ_{NEG} and δ_{POS} can lead to different solutions $\mathcal{S}(o)$.

For example, in a project where sensitivity is a major goal or where confirmatory retests can be quickly and cheaply performed, one could choose $\delta_{\text{NEG}} = 1$ and $\delta_{\text{POS}} = 2$. This would lead to interpretations in $\mathcal{S}(o)$ with potentially more positive pools, resulting in a larger set of putative positives than that obtained using the Hamming distance. Some of the additional putative positives might not be true positives and would therefore be eliminated when they failed to retest, but others could be genuine true positives that had an exceptionally high number of false negative assays, causing them to be missed with the Hamming distance.

This model still has one shortcoming: it assumes that the assays are truly binary and allows only two outcomes, positive or negative. Any information that may have been available with regards to the strength of the signal is lost and cannot be used in the decoding. For some assays this can be significant. For example, in yeast-two-hybrid interactome mapping, the assay readout for true positives can vary widely in intensity, ranging from a strong unmistakable signal to a weakish one that could easily be a false positive.

This situation can be taken into account by using more than two discrete outcomes. For example, consider Ω and δ defined by: $\Omega = \{\text{NONE}, \text{FAINT}, \text{WEAK}, \text{STRONG}\}$, $\mathcal{N} = \{\text{NONE}, \text{FAINT}\}$, $\mathcal{P} = \{\text{WEAK}, \text{STRONG}\}$, $\delta_{\text{NONE}} = 2$, $\delta_{\text{FAINT}} = 1$, $\delta_{\text{WEAK}} = 2$, and $\delta_{\text{STRONG}} = 4$. This model allows four discrete outcomes. NONE is the typical negative signal. The FAINT signal is also a priori negative, but can easily be considered as a false negative ($\delta_{\text{FAINT}} = 1$). WEAK represents a moderate positive signal, and STRONG is reserved for very clear signals that are unlikely to be false positives ($\delta_{\text{STRONG}} = 4$).

As shown, Ω and δ can be used to specify a wide variety of models for the experimental errors.

In the beginning of this section, we gave a simple algorithm for testing whether an interpretation is consistent and identifying the putative positives if it is. Assuming $\mathcal{S}(o)$ is known, this algorithm can be applied to every interpretation of $\mathcal{S}(o)$. When $|\mathcal{S}(o)| > 1$, several strategies can be employed for merging the decoding results of the different nearest consistent interpretations (e.g. intersection, union, majority,...), but it is a matter of policy: in any case identification of $\mathcal{S}(o)$ is required, and clearly constitutes the computational bottleneck.

One could imagine using the following naïve algorithm:

(1) $d = 0$.
(2) Test the consistency of every interpretation at distance d.
(3) If no interpretation was consistent, increment d and go to (2).

However, the number of interpretations at a given distance d increases exponentially with d. In addition, if the set of pools is well chosen, the distance to the nearest consistent interpretation should by and large be proportional to the number of observation errors; otherwise, the number of pools should typically be increased to avoid 'mis-taggings', i.e. erroneous decoding results. Therefore the naïve algorithm cannot be used with realistic error-rates.

One tempting idea would be to use integer linear programming (ILP) methods. Indeed, it is straightforward to express the decoding problem as an ILP problem, which can then be solved

using general software such as GLPK or lp_solve. We successfully applied this approach on small toy examples (100 variables). However, in our hands the ILP solvers' performance degrades rapidly when attempting to scale up to realistic problem sizes: in the specific context of the decoding problem, the ILP approach does not appear more powerful than the naïve algorithm, leaving us in need of a better solution.

3 METHODS

In this section, we present the concepts that underlie the *interpool* algorithm. The notions of 'conflicting variables' and 'conflicting pools' are introduced. We then define the 'score' of a set of conflicting negative pools, and the 'closure' of a set of variables. Finally, we show that the decoding problem can be solved by finding the set of maximal-scoring closures of conflicting variables.

For clarity, all definitions and theorems are presented in the simplest instance of the decoding problem, i.e. where $\Omega = \{\text{POS}, \text{NEG}\}$ and $\delta_{\text{POS}} = \delta_{\text{NEG}} = 1$. They can easily be generalized to arbitrary Ω and δ, and all proofs still hold. Proofs are provided in the supplementary materials.

3.1 Notations

$\forall v \in \mathcal{V}$, we note $\pi(v) \triangleq \{p \in \mathcal{D} \mid v \in p\}$ the set of pools that contain v. Given an interpretation I, we note $N(\text{I})$ and $P(\text{I})$ the set of negative and positive pools (in I), respectively. Similarly, we note $N(o)$ and $P(o)$ the sets of negative and positive pools in an observation o. Using these notations, the distance between an observation o and an interpretation I is:

$$\delta(o, \text{I}) = \left|N(o) \cap P(\text{I})\right| + \left|P(o) \cap N(\text{I})\right|.$$

The first term represents the distance induced by the pools observed negative but interpreted positive (i.e. interpreted as false negatives), and the second term accounts for the distance induced by the pools interpreted as false positives.

3.2 Conflicting variables, conflicting pools

Given an observation o, the partitioning of Ω into $\mathcal{N}$ and $\mathcal{P}$ obviously splits the set of pools into two categories: negative and positive, noted $N(o)$ and $P(o)$ as stated above. However, a more thorough analysis reveals that each category can further be partitioned into two subclasses, which we call 'conflicting' and 'non-conflicting'. These classes, and the underlying notion of 'conflicting variables', can be described as follows.

- A positive pool is non-conflicting in o, if it contains at least one variable that appears only in positive pools. Otherwise, it is conflicting.
- A variable is conflicting in o if it appears in at least one conflicting positive pool.
- A negative pool is non-conflicting in o, if it does not contain any conflicting variables. Otherwise, it is conflicting.

Let us now define these classes formally.

DEFINITION 6. *Conflicting pools and variables. Let o be an observation. The classes of non-conflicting and conflicting positive pools (respectively negative pools) in o, noted* $P_{\bar{c}}(o)$ *and* $\text{P}_{\text{c}}(o)$ *[respectively* $N_{\bar{c}}(o)$ *and* $N_{\text{c}}(o)$*], are:*

$$P_{\bar{c}}(o) \triangleq \left\{p \in P(o) \mid \exists v \in p,\ \pi(v) \subset P(o)\right\},$$
$$P_{c}(o) \triangleq \left\{p \in P(o) \mid \forall v \in p,\ \pi(v) \cap N(o) \neq \emptyset\right\},$$
$$N_{\bar{c}}(o) \triangleq \left\{p \in N(o) \mid \forall v \in p,\ \pi(v) \cap P(o) \subset P_{\bar{c}}(o)\right\},$$
$$N_{c}(o) \triangleq \left\{p \in N(o) \mid \exists v \in p,\ \pi(v) \cap P_{c}(o) \neq \emptyset\right\}.$$

The set of conflicting variables $\mathcal{V}_c$ *is*:

$$V_c \triangleq \left\{v \in V \mid \exists p \in P_c(o),\ v \in p\right\}.$$

These definitions can be extended naturally to interpretations. For any interpretation I, clearly $P_{\bar{c}}(\text{I})$, $P_c(\text{I})$, $N_{\bar{c}}(\text{I})$ and $N_c(\text{I})$ form a partition of the set of pools. It is easy to see that:

$$P_c(\text{I}) = \emptyset \ \Leftrightarrow\ P_{\bar{c}}(\text{I}) = P(\text{I}) \ \Leftrightarrow\ N_c(\text{I}) = \emptyset$$
$$\Leftrightarrow\ N_{\bar{c}}(\text{I}) = N(\text{I}) \Leftrightarrow\ \text{I} \in \mathcal{C}.$$

In particular, an interpretation is consistent if and only if it has no conflicting positive pools. It follows that given an observation o, $\mathcal{S}(o)$ is the set of interpretations I such that $P_c(\text{I}) = \emptyset$ and $\delta(o, \text{I}) = \Delta(o)$.

The following Lemma shows that any pool that is non-conflicting in o keeps its (canonical) value in any nearest consistent interpretation.

LEMMA 1. *Let o be an observation and* $\text{I} \in \mathcal{S}(o)$*, then:*

$$P_{\bar{c}}(o) \subset P(\text{I}), \text{ and } N_{\bar{c}}(o) \subset N(\text{I}).$$

This leads us to propose the following strategy: given an observation o, start off with its canonical interpretation, and empty $P_c(o)$ as 'cheaply' as possible, by changing the values of conflicting pools. Changing conflicting positive pools to negative can obviously contribute to this goal, but changing conflicting negative pools to positive may also do so. Indeed, it can result in conflicting positive pools becoming non-conflicting, as explicited below.

3.3 Score of a set of negative conflicting pools

DEFINITION 7. *Consistent interpretation associated to a subset of* $N_c(o)$*. Let o be an observation. For every* $\text{N} \subset N_c(o)$*, the interpretation associated to* N*, noted* $\iota(\text{N})$*, is defined by:*

$$P(\iota(\text{N})) = \{p \in P(o) \cup \text{N} \mid \exists v \in p, \pi(v) \subset P(o) \cup \text{N}\}.$$

One way to understand this is the following.
Consider the interpretation I' defined by $P(\text{I}') = P(o) \cup \text{N}$. By definition, $P_{\bar{c}}(\text{I}') = \left\{p \in P(\text{I}') \mid \exists v \in p,\ \ \pi(v) \subset P(\text{I}')\right\}$. This is precisely $P(\iota(\text{N}))$. Therefore, $\iota(\text{N})$ is the interpretation where, starting from I', all conflicting positive pools [i.e. $P_c(\text{I}')$] are changed to negative. It is easy to verify that $P(\iota(\text{N})) = P_{\bar{c}}(\iota(\text{N}))$, hence $\iota(\text{N}) \in \mathcal{C}$: the interpretation associated to any $\text{N} \subset N_c(o)$ is consistent. The following Lemma shows that, somewhat reciprocally, every nearest consistent interpretation is associated to some subset of $N_c(o)$.

LEMMA 2. *Let* o *be an observation.*
$\forall\ \text{I} \in \mathcal{S}(o)$*,* $N(o) \cap P(\text{I}) \subset N_{\text{c}}(o)$ and $\text{I} = \iota(N(o) \cap P(\text{I}))$.

DEFINITION 8. *Score of a subset of* $N_c(o)$*.*
Let o be an observation. $\forall\ \text{N} \subset N_c(o)$*, we define the score of* N *as:*

$$\sigma(\text{N}) \triangleq \left|P_c(o) \cap P(\iota(\text{N}))\right| - |\text{N}|.$$

For example, $\iota(\emptyset)$ is such that $P(\iota(\emptyset)) = P_{\bar{c}}(o)$: it is the interpretation where, starting from o's canonical interpretation, every positive conflicting pool is changed to negative. Clearly, $\iota(\emptyset)$ is consistent, and $\sigma(\emptyset) = 0$.

Generally, the score of N can be interpreted as follows. Changing all the pools in N to positive has a cost: it induces a distance $|\text{N}|$. However, this leads to some conflicting positive pools becoming non-conflicting, namely $P_c(o) \cap P(\iota(\text{N}))$: these pools remain positive in $\iota(\text{N})$, whereas they must be changed to negative in $\iota(\emptyset)$. $\sigma(\text{N})$ is therefore the difference between what is gained and what is paid, when considering $\iota(\text{N})$ as a possible solution [i.e. as an element of $\mathcal{S}(o)$] and $\iota(\emptyset)$ as the reference point.

The distance between o and $\iota(\emptyset)$ is $\delta(o, \iota(\emptyset)) = |P_c(o)|$. In the case of an arbitrary N, the relationship between $\sigma(\text{N})$ and $\delta(o, \iota(\text{N}))$ is explicited in Lemma 3, leading immediately to Lemma 4.

LEMMA 3. *Let o be an observation.*

$$\forall \mathrm{N} \subset N_c(o),\ \sigma(\mathrm{N}) + \delta(o, \iota(\mathrm{N})) \leq \left|P_c(o)\right|,$$

with equality if and only if $\mathrm{N} \subset P(\iota(\mathrm{N}))$.

LEMMA 4. *Let o be an observation.*

$$\forall \mathrm{N} \subset N_c(o),\ \sigma(\mathrm{N}) \leq \left|P_c(o)\right| - \Delta(o),$$

with equality if and only if $\iota(\mathrm{N}) \in \mathcal{S}(o)$ and $\mathrm{N} \subset P(\iota(\mathrm{N}))$.

In conjunction with Lemma 2, we obtain:

$$S(o) = \{\, \iota(\mathrm{N}) \mid \mathrm{N} \subset N_c(o)\ ,\ \sigma(\mathrm{N}) \text{ maximal} \,\}.$$

$\mathcal{S}(o)$ can therefore be identified by finding the highest-scoring subsets of $N_c(o)$. In addition, Lemma 4 shows that this search can be restricted to the subsets $\mathrm{N} \subset N_c(o)$ that satisfy $\mathrm{N} \subset P(\iota(\mathrm{N}))$. In fact, this condition simply states that all pools of N must be positive in $\iota(\mathrm{N})$. Given Definition 7 and the ensuing remark, this seems natural. Indeed, if $p_1 \in \mathrm{N}$ is such that $p_1 \in N(\iota(\mathrm{N}))$, then one can consider $\mathrm{N}' = \mathrm{N} \setminus \{p_1\}$, and obviously $\iota(\mathrm{N}') = \iota(\mathrm{N})$: p_1 becomes positive in I' defined by $P(\mathrm{I}') = P(o) \cup \mathrm{N}$, but this is useless since p_1 gets changed back to negative in $\iota(\mathrm{N})$.

In the following section, we see that this condition can be expressed simply in terms of closures of variables.

3.4 Closure of a set of conflicting variables

DEFINITION 9. *Closure of a set of conflicting variables.*
Let V_1 be a set of conflicting variables. The closure of V_1 in a subset of $\mathcal{D}$ is the set of pools of the subset that contain some variable of V_1. In particular, the closures of V_1 in $P_c(o)$ and in $N_c(o)$, denoted $\pi_p(V_1)$ and $\pi_{\mathrm{n}}(V_1)$, are:

$$\pi_p(V_1) \triangleq \bigcup_{v \in V_1} \pi(v) \cap P_c(o) \qquad \text{and}$$

$$\pi_n(V_1) \triangleq \bigcup_{v \in V_1} \pi(v) \cap N_c(o).$$

LEMMA 5. *Let o be an observation, and $\mathrm{N} \subset N_c(o)$ such that $\sigma(\mathrm{N})$ is maximal. Then:*

$$\exists V_1 \subset V_c,\ \mathrm{N} = \pi_n(V_1).$$

Combined with the previous section's conclusion, we finally obtain the following reformulation of the decoding problem.

THEOREM 1. *Let* o *be an observation.*

$$S(o) = \left\{\, \iota\big(\pi_n(V_1)\big) \mid V_1 \subset V_c\ ,\ \sigma\big(\pi_n(V_1)\big) \text{ maximal} \,\right\}.$$

The decoding problem can therefore be solved by finding the set(s) of conflicting variables whose closure(s) in $N_c(o)$ has/have the highest score. This presentation of the problem has the advantage that for any set of conflicting variables V_1, the score of its closure can be easily calculated: $\sigma(\pi_{\mathrm{n}}(V_1)) = |\pi_p(V_1)| - |\pi_n(V_1)|$, provided that $V_1 = \{v \in V_c \mid \pi_n(\{v\}) \subset \pi_n(V_1)\}$. This condition simply states that V_1 should include as many conflicting variables as possible, as long as they do not change the set $\pi_n(V_1)$. If the condition is not satisfied, $|\pi_p(V_1)| - |\pi_n(V_1)|$ underestimates $\sigma(\pi_n(V_1))$; the correct score is calculated by considering V_2, the largest superset of V_1 such that $\pi_n(V_2) = \pi_n(V_1)$.

4 ALGORITHM

In this section, we describe the *interpool* algorithm for solving the decoding problem described in Definition 5. This algorithm relies on Theorem 1: it identifies all maximal-scoring closures (in $N_c(o)$) of sets of conflicting variables.

As in Section 3, for the sake of clarity results are presented in the simplest instance of the decoding problem. Their analogs remain valid in the general framework and have been implemented, but they require unwieldy notations. For example, $|P_c(o)|$ becomes $\Sigma_{\omega \in \mathcal{P}}\ \delta_\omega \cdot |\{p \in P_c(o) \mid o(p) = \omega\}|$.

4.1 The search space

The goal is to find all sets of conflicting variables $V \subset \mathcal{V}_c$ such that $\sigma(\pi_n(V))$ is maximal. The algorithm follows a branch-and-bound strategy. The general idea is to build V by adding conflicting variables one at a time. The search space is a tree where each node represents a set of conflicting variables: the root is the empty set, and the children of a node V_1 represent the sets $V_1 \cup \{v\}$, where v is not in V_1 and not in any elder brother of V_1 or of any of its ancestors.

We use the term unit closure to represent the closure $\pi_n(\{v\})$ of a single conflicting variable v. Before beginning the search, the values of $\pi_n(\{v\}), \pi_p(\{v\}), |\pi_n(\{v\})|$ and $\sigma(\pi_n(\{v\}))$ are pre-calculated for every $v \in \mathcal{V}_c$. These quadruplets, which represent unit closures and their associated information, are referred to as unit structures.

The search tree is explored following a depth-first strategy, and the variables are initially sorted by decreasing score (of the corresponding unit structure). This leads to the quick identification of some high-scoring—though not necessarily optimal—closures, which prove valuable for pruning as discussed in Section 4.2. Internal nodes are considered as virtual observations, derived from the real observation but where some choices have already been made: namely, that the selected conflicting variables are actually positive, and that any conflicting negative pool that contains them is a false negative. At each step leafwards, the unit structures are updated by 'substracting' the selected variable's unit structure from them. This process is performed by the function substractFromUnits, and detailed in Section 4.3. As a result, internal nodes along with their 'local' unit structures can be dealt with—and pruned where appropriate—in the same way as the root node, with a small adjustment to take into account the selected closures. For example, the score of each child node can be trivially calculated (or underestimated as discussed following Theorem 1, but this is easily corrected) by simply adding the corresponding unit structure's local score to the current score. In addition, the search in each sub-tree is performed by decreasing order of local score instead of the less relevant initial score. Finally, although the search space remains huge, the following propositions can be used to prune large sub-trees, resulting in an efficient exact algorithm.

4.2 Pruning criterion

Propositions 1 and 2 can be combined to provide an upper bound for the score of any remaining descendant of the current node. When this upper bound is smaller than the current best score, the corresponding sub-trees can be ignored and pruned. These propositions are effective even early in the search, because high-scoring closures are found fast due to the search strategy, as discussed in Section 4.1.

NOTATION. *In this sub-section, we use the following notations to represent instrinsic characteristics of the pooling design* $\mathcal{D}$*:*

- k *is the pooling design's maximal redundancy* (*maximum number of pools that can contain any single variable*).
- Γ *is the maximal co-occurence of variables in* $\mathcal{D}$*, i.e. the maximum number of pools that can contain any two variables.*

PROPOSITION 1. *Let* $q \in \mathbb{N}^*$*, and* $\{\mathrm{N}_i\}_{i=1,\ldots,q}$ *be the* q *highest-scoring unit closures. Consider a set of unit closures* $\{\mathrm{N}'_i\}_{i=1,\ldots,q}$ *corresponding to conflicting variables* $\{v'_i\}_{i=1,\ldots,q}$ *satisfying the following condition:* $\forall v \in V_c \setminus \{v'_i\}_{i=1,\ldots,q},\ \pi_n\{v\} \not\subset \bigcup_{i=1}^{q} \mathrm{N}'_i$*. Then:*

$$\sigma\left(\bigcup_{i=1}^{q} \mathrm{N}'_i\right) \leq \sum_{i=1}^{q} \Big(\sigma(\mathrm{N}_i) + \min\big((i-1)\Gamma, k-1\big)\Big).$$

PROPOSITION 2. *Let* $q \in \mathbb{N}^*$*, and* $\{\mathrm{N}_i\}_{i=1,\ldots,q}$ *be the* q *smallest unit closures sorted by increasing size. Then for any set of* $q' \geq q$ *unit closures* $\{\mathrm{N}'_i\}_{i=1,\ldots,q'}$*:*

$$\sigma\left(\bigcup_{i=1}^{q'} \mathrm{N}'_i\right) \leq |P_c(o)| - \sum_{i=1}^{q} \Big(|\mathrm{N}_i| - \min\big((q-i)\Gamma, |\mathrm{N}_i|\big)\Big).$$

Note that the upper bound in proposition 1 tends to increase with q, while that in Proposition 2 can only decrease. They can be used conjointly to obtain a powerful pruning criterion. Indeed, noting α the current best score (adjusted by substracting the current node's score), one can apply the following algorithm:

(1) Find the smallest q such that

$$\sum_{i=1}^{q} \Big(\sigma(\mathrm{N}_i) + \min\big((i-1)\Gamma, k-1\big)\Big) \geq \alpha,$$

where $\{\mathrm{N}_i\}_{i=1,\ldots,q}$ are the q highest-scoring unit closures.

(2) Consider now the q smallest unit closures $\{\mathrm{N}'_i\}_{i=1,\ldots,q}$ sorted by increasing size. If

$$\sum_{i=1}^{q} \Big(|\mathrm{N}'_i| - \min\big((q-i)\Gamma, |\mathrm{N}'_i|\big)\Big) > |P_c(o)| - \alpha,$$

every remaining descendant of the current node can be pruned.

This algorithm's correctness, which results from the two propositions, is proved in the supplementary materials. From the point of view of the search tree, the algorithm anticipates several moves in advance: the first step bounds the score of any node less than q generations leafwards, while the second step does so for nodes at least q generations away.

4.3 The *interpool* algorithm

A rough description of the core of *interpool* is the recursive algorithm presented in Figure 1.

Before the initial call to findBest, all conflicting pools and variables are identified. For each conflicting variable, its unit structure is determined: this is used to initialize *units*. The algorithm is intially called with the empty structure (defined by $\pi_n(\emptyset) = \emptyset$, $\pi_p(\emptyset) = \emptyset$ and $\sigma(\emptyset) = 0$) serving as both *previous* and *best*. Indeed, it constitutes a possible solution corresponding to the interpretation where every positive conflicting pool becomes negative.

```
findBest(units, best, previous) {
  sort units by decreasing score;
  sort units by increasing size;
  for each u ∈ units (in order of decreasing score) {
    if (pruning algorithm succeeds)
      break;
    units = units \ u, current = previous ∪ u;
    newUnits = substractFromUnits(units, u, current);
    updateBest(best, current);
    findBest(newUnits, best, current); }}
```

Fig. 1. The core of the *interpool* algorithm, described in pseudo-C. *units* is the list of unit structures, *best* stores the highest-scoring structures found up to now, and *previous* holds the structure examined in this node's father.

Upon being called, the unit structures *units* are sorted by decreasing score as well as by increasing size. In addition to guiding the search, these orderings are used within the pruning algorithm as described in Section 4.2. Each unit structure is then selected successively, by order of decreasing score. If the algorithm derived from Propositions 1 and 2 authorizes pruning, the function returns immediately: this skips the currently selected unit structure and its descendance, but also every unit structure that remained to be selected in *units*. Otherwise, the current unit structure u is removed from *units* and merged with the running structure *previous* to obtain *current* (i.e. the closures are unioned and the scores and sizes are added). It is also 'substracted' from each unit structure of *units* to obtain *newUnits*. More precisely, substractFromUnits substracts each component (π_n and π_p) of u from the remaining unit structures, and updates their scores and sizes accordingly. In addition, it discards unit structures if:

(1) the resulting unit structure no longer has any conflicting positive pools, i.e. the variable isn't conflicting anymore;

(2) the resulting unit structure no longer has any conflicting negative pools, in which case its former conflicting positive pools are added to π_p of *current* beforehand.

The latter step results in the score of *current* being exact, rather than under-estimated as discussed following Theorem 1. This score is then compared to the current best score by updateBest, and *best* is updated if required. Finally, the recursive call occurs. When the initial call returns, *best* holds the highest-scoring structures.

Determining the average complexity of interpool is hard, as is often the case for combinatorial optimization algorithms. Indeed, the search tree is dynamically built and pruned as the search progresses, and this process cannot be easily modeled. In addition, as discussed in Section 6, the performance of interpool does not depend so much on the total number of

Table 1. Simulation results: STD(940;13;13), false positive rate 10%

Positives	FNR%	TPs missed	Retests	Simulations	Time
2	10	0	2.26	10 000	1 min
2	20	0	2.26	10 000	1 min
2	30	1.2%	2.27	10 000	4 min
3	10	0	3.57	10 000	4 min
3	20	0.4%	3.58	10 000	33 min
3	30	3.4%	3.60	10 000	2 h
4	10	0	5.06	10 000	32 min
4	20	1.0%	5.11	10 000	10 h 39 min
4	30	6.2%	5.26	7500	2 days 11 h
5	10	0.1%	6.71	10 000	4 h
5	20	1.7%	6.94	1000	12 h 47 min
5	30	12.9%	7.88	300	3 days 10 h

Positives: number of simulated positives. FNR: false negative rate of the individual assays. TPs missed: fraction of true positives that are not recovered. Retests: number of variables decoded as positive or ambiguous, that must be retested (this includes the recovered true positives). TPs missed and Retests are the upper bounds of the 95% confidence interval for the mean. Simulations: number of simulations performed in the run. Time: total real time taken by the run, on a 2.13 GHz Intel Core 2 Duo GNU/linux system.

variables and pools. Instead, it depends mainly on how 'comfortable' the pooling design is, with respect to the number of positives and errors. This is difficult to quantify, hence the choice of the input size to use in a complexity analysis is unclear. However, Table 1, with run-times for a variety of realistic instances, provides a useful indication of how interpool performs.

5 IMPLEMENTATION

The *interpool* algorithm has been implemented in C, and is freely available under the terms of the GNU General Public Licence (GPL). It builds cleanly and has been tested on several hardware and software combinations, including various GNU/linux i386 and x86_64 setups, SunOS 5.9 and Cygwin. It can be downloaded from our web page (http://www-timc.imag.fr/Nicolas.Thierry-Mieg/).

The implementation allows up to four discrete outcomes, two of which are interpreted as positive and two as negative: this corresponds to the instance $\Omega = \{\text{NONE}, \text{FAINT}, \text{WEAK}, \text{STRONG}\}$, $\mathcal{N} = \{\text{NONE}, \text{FAINT}\}$, and $\mathcal{P} = \{\text{WEAK}, \text{STRONG}\}$ presented in Section 2. The distance δ is set by the user. Additional discrete outcomes could be allowed with a little work, although we have never felt the need in our applications: changing the costs in δ provides sufficient flexibility.

The code has been heavily optimised for speed. For example, the data structures have been chosen so that all bottleneck calculations are implemented as bitwise operations. This provides high efficiency, and additionally allows to take full advantage of modern 64-bit architectures. Indeed, on an Intel Core 2 Duo CPU the *interpool* implementation is almost twice as fast when compiled in 64-bit mode as it is in 32-bit mode (produced by compiling with the -m32 switch to gcc). Another useful trick is that when entering findBest, the unit structures *units* are not completely sorted. Instead, we only identify and sort a small number of top-ranking structures, and later extend the sorted lists if necessary. In practice, extending is relatively rarely needed, due to pruning fairly early in the loop.

In addition to the *interpool* algorithm proper, the package includes several independant implementations of simpler decoding algorithms that are used for cross-validation, as well as a tool for performing simulations. One simulation consists in four steps: (1) randomly pick t variables: the simulated positives; (2) generate the corresponding observation, add noise; (3) interpret the observation, using *interpool*, to obtain a decoded value for each variable; (4) compare the variables' decoded values with their real ones (from step 1) to identify any mistaggings. The simulator's usefulness is illustrated in Section 6.

6 RESULTS AND DISCUSSION

The *interpool* algorithm is essential both before and after the actual experimental work. First, because it is very efficient, it allows to perform large numbers of simulations for choosing the pooling design most appropriate for a given experimental setup. For example, with the Shifted Transversal Design (STD; Thierry-Mieg, 2006b), the goal is to select values for parameters q and k such that the design achieves the desired compromise between number of pools and decoding power—given that this is always a trade-off, since identifying more positives and/or correcting more errors requires a greater number of pools. Second, once the pools have been selected, built and screened, it allows to decode the experimental observations.

Although this latter task is paramount, the former is computationally much more intensive. Indeed, given the sizes of the spaces that are being explored, large numbers of simulations must be performed in order to obtain satisfactory coverage. We illustrate this process in the context of a pilot project we are involved with in collaboration with Marc Vidal and co-workers (Dana Farber Cancer Institute, Boston), where smart-pooling with STD is applied to yeast-two-hybrid interactome mapping.

In this project, we wish to screen some baits against 940 prey proteins. For each bait, we generally expect at most 3 positive preys. Rough estimates of the expected error-rates are in the 5–10% range for false positives and up to 25% for false negatives.

We first performed a series of 'easy' runs of simulations, with wide ranges of values for the STD parameters q and k, and the two least demanding experimental conditions (2 or 3 positives, 5% false positives and 10% false negatives). Each run comprised 10 000 simulations, and we ruled out any design that showed signs of weakness, i.e. failed to systematically identify every simulated positive, or gave rise to more than 0.1 ambiguous variables on average (out of the 940 variables). We also terminated runs that took longer than 2 min to complete the 10 000 simulations, and excluded the corresponding designs. This is justified by our observation that although *interpool* is very fast when the conditions (number of positives and errors) can be comfortably dealt with by the pooling design, its performance degrades when the design can barely or imperfectly cope with the conditions. Therefore, slow runs in the easy conditions are an indication that the design will not be powerful enough in more difficult ones. This was confirmed

by performing small numbers of simulations with increased error rates. This first stage led to the pre-selection of five STD designs.

In a second step, each candidate design was examined in more detail: the number of positives varied between 1 and 5, while the error-rates were set at up to 15% false positives and 30% false negatives. In this way, the behaviour of each design in the case of highly connected baits and/or unexpectedly high error-rates could be studied. The main measure of performance is the fraction of true positives that are not recovered: it represents the false negative rate of the smart-pooling method. Another interesting measure is the number of preys that are decoded as positive or ambiguous, i.e. the retest burden, assuming the strategy is to retest all the candidates individually.

For example, Table 1 shows results obtained with the design STD(940;13;13), using a false positive rate of 10%. This design performs well in most settings, although it begins to miss a non-negligible fraction of positives when there are three or more positives and a 30% false negative rate. Yet even in the hardest setting, the smart-pooling false negative rate is only 12.9%: much less than that of the individual pairwise screen (30%), which requires 940 tests (instead of 169 for this STD design). Notice that in all settings, the retest burden is at most a few more than the number of true positives: most candidates will be genuine positives. In this phase, conditions leading to slow runs were still studied, although the number of simulations was decreased when necessary. This results in the measured means for the two performance criteria being less precise; but since we report the upper bounds of the 95% confidence intervals, which are correspondingly larger, the reported numbers remain valid over-estimates of the true means and can be compared to the other conditions.

As a side note, this data confirms that the guarantee requirement is indeed overkill for practical purposes. For example, identification of three positives with STD(940;13;13) is only guaranteed when there are at most three errors of each type, but the error rates used here when $t = 3$ correspond to 13 false positives and 3, 7 and 10 false negatives (for 10, 20 and 30% respectively). Clearly, the first two conditions are dealt with very well despite 10 excess false positives and up to four extra false negatives. This shows how important it is to perform simulations in order to choose a design.

After comparing similar datasets obtained with the other candidate designs, this one was selected as the best compromise between robustness and size—it has 169 pools, hence fits into two 96-well plates. Based on these results, a pilot experiment was performed in collaboration with Marc Vidal and co-workers, where 100 baits were screened against 940 preys smart-pooled according to STD(940;13;13). This experiment will be reported elsewhere.

7 CONCLUSION

In this article, we provide a clear formalization of the decoding problem and present a deterministic algorithm to solve it. This algorithm is exact, i.e. it always finds the optimal solution, yet proves very efficient. An open-source implementation is freely available. It can be used to perform simulations in order to choose appropriate sets of pools for a given application before carrying out assays. Subsequently, it allows to interpret the experimental results, correcting for false positives and false negatives and identifying the positives.

ACKNOWLEDGEMENTS

We thank Jean Thierry-Mieg, Laurent Trilling, Jean-Louis Roch and Michael Blum for stimulating discussions. This work was funded by a BQR'2003 grant from the Institut National Polytechnique de Grenoble (INPG) to NT.

Conflict of Interest: none declared.

REFERENCES

Barillot,E. *et al.* (1991) Theoretical analysis of library screening using a N-dimensional pooling strategy. *Nucleic Acids Res.*, **19**, 6241–6247.

Bruno,W.J. *et al.* (1995) Efficient pooling designs for library screening. *Genomics*, **26**, 21–30.

Jin,F. *et al.* (2006) A pooling-deconvolution strategy for biological network elucidation. *Nat. Methods*, **3**, 183–189.

Jin,F. *et al.* (2007) A yeast two-hybrid smart-pool-array system for protein-interaction mapping. *Nat. Methods*, **4**, 405–407.

Knill,E. *et al.* (1996) Interpretation of pooling experiments using the Markov chain Monte Carlo method. *J. Comput. Biol.*, **3**, 395–406.

Thierry-Mieg,N. (2006a) Pooling in systems biology becomes smart. *Nat. Methods*, **3**, 161–162.

Thierry-Mieg,N. (2006b) A new pooling strategy for high-throughput screening: the Shifted Transversal Design. *BMC Bioinformatics*, 7:28.

Vermeirssen,V. *et al.* (2007) Matrix and Steiner-triple-system smart pooling assays for high-performance transcription regulatory network mapping. *Nat. Methods*, **4**, 659–664.

Shifted Transversal Design smart-pooling for high coverage interactome mapping

Xiaofeng Xin, Jean-François Rual, Tomoko Hirozane-Kishikawa, et al.

Genome Res. 2009 19: 1262-1269 originally published online May 15, 2009
Access the most recent version at doi:10.1101/gr.090019.108

Shifted Transversal Design smart-pooling for high coverage interactome mapping

Xiaofeng Xin,[1,4] Jean-François Rual,[2,5] Tomoko Hirozane-Kishikawa,[2] David E. Hill,[2] Marc Vidal,[2,6] Charles Boone,[1,6] and Nicolas Thierry-Mieg[3,4,6]

[1]*Banting and Best Department of Medical Research and Department of Molecular Genetics, Terrence Donnelly Centre for Cellular and Biomolecular Research, University of Toronto, Toronto, Ontario M5S 3E1, Canada;* [2]*Center for Cancer Systems Biology (CCSB), and Department of Cancer Biology, Dana-Farber Cancer Institute and Department of Genetics, Harvard Medical School, Boston, Massachusetts 02115, USA;* [3]*Laboratoire TIMC-IMAG, TIMB, CNRS UMR5525, Faculte de Medecine, 38706 La Tronche Cedex, France*

"Smart-pooling," in which test reagents are multiplexed in a highly redundant manner, is a promising strategy for achieving high efficiency, sensitivity, and specificity in systems-level projects. However, previous applications relied on low redundancy designs that do not leverage the full potential of smart-pooling, and more powerful theoretical constructions, such as the Shifted Transversal Design (STD), lack experimental validation. Here we evaluate STD smart-pooling in yeast two-hybrid (Y2H) interactome mapping. We employed two STD designs and two established methods to perform ORFeome-wide Y2H screens with 12 baits. We found that STD pooling achieves similar levels of sensitivity and specificity as one-on-one array-based Y2H, while the costs and workloads are divided by three. The screening-sequencing approach is the most cost- and labor-efficient, yet STD identifies about twofold more interactions. Screening-sequencing remains an appropriate method for quickly producing low-coverage interactomes, while STD pooling appears as the method of choice for obtaining maps with higher coverage.

[Supplemental material is available online at www.genome.org. The protein interactions from this publication have been submitted to the IMEx (http://imex.sf.net) Consortium through IntAct (PMID 17145710) and assigned the identifier IM-11695.]

Genome projects have enabled the development of a variety of large-scale functional genomics and proteomics projects. Some aim at identifying relatively rare events, such as mapping of binary protein–protein interactions (PPIs), protein–DNA interactions, or genetic interactions (for example, Yu et al. 2008, Deplancke et al. 2006, and Tong et al. 2004, respectively). These projects typically face three issues: reducing the cost and the number of assays (efficiency), recognizing false-positives that reflect technical artifacts (specificity), and avoiding false-negatives (sensitivity). Performing individual tests multiple times remains the gold standard for data quality but is often prohibitively costly and time-consuming. A frequently used alternative consists in assaying pools and then identifying the positives in a second step. For example, in the yeast two-hybrid (Y2H) screening-sequencing approach (Screen-Seq), first a set of preys are pooled and screened for interaction with a specific bait, then the positive clones are sequenced to identify the interactions (Rual et al. 2005). Alternatively, if a positive is detected in a pool, then all the constituents of the positive pool can be retested individually (Zhong et al. 2003; Stelzl et al. 2005). These methods improve the efficiency, and false-positives can be limited by subsequent experiments such as stringent retests, but false-negatives in the initial screen cannot be recovered.

[4]These authors contributed equally to this work.
[5]Present address: Department of Cell Biology, Harvard Medical School, Boston, MA 02115, USA.
[6]Corresponding authors.
E-mail Nicolas.Thierry-Mieg@imag.fr; fax +33-456-520-055.
E-mail charlie.boone@utoronto.ca; fax (416) 978-8287.
E-mail marc_vidal@dfci.harvard.edu; fax (617) 632-5739.
Article published online before print. Article and publication date are at http://www.genome.org/cgi/doi/10.1101/gr.090019.108.

Another method called "smart-pooling" (or "group testing") aims to further increase efficiency, accuracy, and coverage in high-throughput screening projects. Smart-pooling has been used for screening clone libraries (Bruno et al. 1995) and was recently employed for Y2H (Jin et al. 2006, 2007) and yeast one-hybrid screens (Vermeirssen et al. 2007). It consists of assaying well-chosen pools of "items" (for example preys in Y2H) such that each item is present in several pools, hence tested several times (Thierry-Mieg 2006b). The goal is to construct the pools so that the positive items can be identified from the pattern of positive pools, even despite the occurrence of false-positive and false-negative pools.

Central to the smart-pooling procedure is the choice of the pooling design, and key parameters are both the "redundancy" and the "extra redundancy" of the design. We call "redundancy" the number of pools that contain any given item. Part of the redundancy is necessary to identify a single positive item, and the remaining "extra redundancy" allows the system to deal with noise (false-positive and false-negative pools) and multiple positive items (within a particular batch of pools).

Previous smart-pooling systems biology studies (Jin et al. 2006, 2007; Vermeirssen et al. 2007) have established the experimental feasibility of the smart-pooling concept in this field, but they have relied on designs that do not leverage the method's full potential. The PI-deconvolution approach ("pooling with imaginary tags") (Jin et al. 2006, 2007) uses a variant of the classic grid design, where items are arrayed on an imaginary grid and a pool is constructed for each row and each column. Specifically, in PI deconvolution the grid is extended to N dimensions but restricted to a side length of 2 (for example with N = 3 it becomes a $2 \times 2 \times 2$ cube, and each pool is defined by one of the six possible $2 \times 2 \times 1$ slices). From a theoretical point of view, this design can be improved at two levels. First, it has a single degree of freedom:

19:1262–1269 © 2009 by Cold Spring Harbor Laboratory Press; ISSN 1088-9051/09; www.genome.org

Choosing the redundancy imposes the pool size and the number of preys per batch, whereas one would like to set these three characteristics independently. Second, all of its redundancy is required for identifying a single positive in a noiseless experiment, leaving no extra redundancy to deal with multiple positives within a batch or correct for noise. Consequently, decoding is highly ambiguous when multiple positives, false-positives, or false-negatives occur. The second study (Vermeirssen et al. 2007) relied on a more sophisticated design derived from a Steiner system. The design has a redundancy of 3, and two pools uniquely identify a prey, leaving an extra redundancy of 1. As shown (Vermeirssen et al. 2007), this provides improved performance compared to PI deconvolution, but the noise-correction capabilities remain modest when multiple positives occur. The authors overcame this limitation by resorting to sequencing for confirmation or identification of positives, and by "uniplexing" known highly connected transcription factors: these were excluded from the smart-pool design and were tested individually instead, thus reducing the occurrence of multiple positives in a smart-pool batch. This is a useful strategy, but as a consequence it is difficult to interpret the obtained results in terms of success of the smart-pooling per se.

Other theoretical pooling designs that offer higher extra redundancy have been described (for review, see Thierry-Mieg 2006a) but lack experimental validation. In particular, we previously proposed a powerful and flexible algorithm for designing smart-pools: the Shifted Transversal Design (STD) (Thierry-Mieg 2006a). STD was shown to significantly outperform other published combinatorial designs in terms of flexibility and/or efficiency under a standard combinatorial model (the so-called "guarantee requirement," where given bounds on the numbers of positives and erroneous observations, i.e., false-positives and false-negatives, a design must guarantee the identification of all positives). In STD, a large redundancy can be chosen and the extra redundancy is maximized, therefore providing high noise-correction capabilities. However, this power comes at a price: Despite its clean mathematical construction, the design is complex and difficult to visualize. In addition, interpreting experimental results is straightforward with simpler designs whose noise-correction abilities are intrinsically limited (Jin et al. 2006, 2007; Vermeirssen et al. 2007), but it becomes a difficult computational problem with highly redundant designs such as STD. Recently, this has been addressed by developing a new exact and efficient algorithm for interpreting smart-pooling results: interpool (Thierry-Mieg and Bailly 2008).

Here we experimentally evaluate the STD-based smart-pooling method in the context of interactome mapping by Y2H. We screened 12 *Caenorhabditis elegans* SH3 domains as baits against a *C. elegans* ORFeome library comprising 12,675 preys. We employed two STD designs adapted to different array densities, as well as the labor-intensive one-on-one array-based Y2H method (Uetz et al. 2000) (referred to as 1-on-1 hereafter). Additionally, six of these 12 baits were screened with the well-established Screen-Seq approach (Rual et al. 2005). All screens were performed with two repeats, and every positive from each method underwent pairwise retest in quadruplicate. Since all experiments in this study used the same reagents, false-negatives mainly contribute to "sampling sensitivity" in the recently proposed framework (Venkatesan et al. 2009), i.e., they should be identifiable by every method, given enough repeats. Our results show that STD Y2H is highly specific. Compared with 1-on-1, STD is much more cost- and labor-efficient, yet it remains very competitive in terms of sensitivity. While the Screen-Seq method is the most efficient in terms of costs and workload, STD Y2H appears approximately twice as sensitive. STD smart-pooling emerges as a method of choice for obtaining high-coverage interactomes, and could prove effective in a wide range of high-throughput experiments.

Results

Building STD pools for the worm activation domain ORFeome library

To study the application of STD pooling in proteome-scale Y2H, we assembled a set of reagents for array-based Y2H analysis (Uetz et al. 2000). Our prey array consists of 12,675 activation domain (AD) proteins represented in *C. elegans* ORFeomes v1.1 and v3.1 (Reboul et al. 2003; Lamesch et al. 2004). The baits consisted of 12 worm SH3 domains (Supplemental Table 1), a class of peptide recognition modules that often mediate PPIs by binding proline-rich peptide sequences (Ren et al. 1993; Tong et al. 2002).

We built and tested two STD designs with different pool sizes, adapted to different array densities. For any pooling design, the pool size is a major parameter. For example, larger pools improve the efficiency because fewer pools are required, but this may compromise the sensitivity due to dilution of the AD ORFs within the pools. Before designing the STD arrays, we performed dilution tests at two different densities, 384 and 1536 spots per plate, in order to identify the largest pool sizes that enable detection of positive controls (Supplemental Fig. 1; Supplemental Table 2). In the less dense 384 format, more yeast diploids can be obtained and the yeast colonies can grow larger, enabling more sensitive Y2H analysis, and thus allowing a larger pool size. In conjunction with a preliminary pilot experiment (Supplemental Note; Supplemental Fig. 2) and further simulations performed with interpool (data not shown), these initial tests led us to choose pool sizes of 78 for the 384 format and 26 for the 1536-format arrays.

To limit the cost of building the STD pools and to increase their flexibility, we took advantage of inherent STD symmetries by designing and building small intermediary micropools. In Figure 1, a simple example illustrates this process: 18 preys are pooled according to a small STD design. Initially, the 18 preys are split into two groups of nine preys (groups A and B), and each group is pooled independently according to its corresponding STD subdesign to obtain two sets of micropools (sets A and B). Each micropool contains three different preys (pool size of 3), and each prey is contained in three different micropools (redundancy of 3), which form this prey's unique signature. In fact, any two micropools are sufficient to uniquely identify a prey, so these micropools have an extra redundancy of 1. Finally, each pair of same-numbered micropools from sets A and B are superposed to obtain one batch of STD pools (p1–p9). These STD pools still possess a redundancy of 3, but their pool size is now 6, and the nine STD pools accommodate all 18 preys. Each prey still has a unique signature, although the extra redundancy is now 0 because all three pools are required to identify each prey uniquely.

We built the worm STD pools in a similar manner but on a higher scale (Figs. 2, 3). The prey library, which contains 12,675 unique AD ORFs, was conceptually split into 75 groups of 169 preys ($75 \times 169 = 12{,}675$). Each group was STD-pooled independently to obtain a set of 169 worm micropools containing 13 preys each (micropool size: 13). These micropools possess a redundancy of 13, including an extra redundancy of 11. All 12 sets of micropools were built according to subdesigns of a larger STD

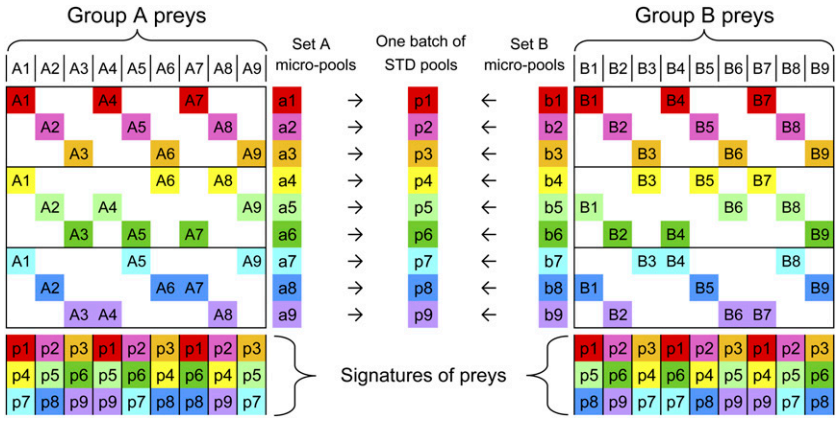

Figure 1. A simple STD pooling design. Two groups of nine preys (group A: A1–A9 and group B: B1–B9) are separately pooled into nine color-coded micropools (set A: a1–a9 and set B: b1–b9), according to subdesigns of a larger STD design (which can accommodate all 18 preys). Each micropool contains three preys (e.g., a1 contains A1, A4, and A7), and each prey is present in a unique combination of three micropools (a unique signature, e.g., a1|a5|a9 for prey A4). The pairs of same-numbered micropools from sets A and B can then be superposed to generate STD pools (one batch: p1–p9), each containing six preys, and each prey still has a unique signature in the STD pools.

design, so that the micropools are superposable to generate larger STD pools (see Fig. 2 and Methods). Based on the chosen 1536- and 384-format pool sizes, the micropools were either superposed in pairs (as shown in Fig. 2), to produce the STD-1536 pools containing 26 preys per pool, or superposed in sextuplets, to generate the STD-384 pools with 78 preys per pool. In the resulting STD pooling designs, the 12,675 preys are either split into 40 batches of STD-1536 pools with up to 338 preys per batch, or 13 batches of STD-384 pools with up to 1014 preys per batch. The STD-1536 and STD-384 batches each contain 169 pools. All batches within an STD design are arrayed as colonies on a series of plates, but the batches are disjoint and decoded independently. Both designs possess an extra redundancy of 10, which provides high noise-correction capabilities.

C. elegans ORFeome-wide experiments

We screened both STD-1536 and STD-384 with the 12 selected worm SH3 domains (Supplemental Table 1). In order to thoroughly evaluate the STD method, we also screened the same baits using the 1-on-1 Y2H approach (Uetz et al. 2000), where each prey is present individually in duplicate on the array in 1536 format. In addition, six of the baits were screened following the established Screen-Seq protocol (Rual et al. 2005). Finally, to address whether high plate density affected the STD method, we screened the three most connected baits against STD-SL, an STD array with small batch and pool size and low density (namely the STD-1536 pools assembled in 384 format instead of 1536). All screens were performed twice to evaluate each method's repeatability. Figure 3 compares the steps of the Y2H approaches used in this study.

Each method produced a list of candidate PPIs. All candidate PPIs underwent pairwise retest in quadruplicate, leading to the definition of three classes of hits: core, the strong and reproducible positives; FP, false-positive, when the retest fails; and noncore, when the retest results are unclear (e.g., quadruplicate retesting results in two positives and two negatives, see Methods for details). The noncore data set is small (Supplemental Table 3) and was not included in our subsequent analyses. Our classification relies only on the retest results and is independent of the hit's origin (number of detection methods, confidence scores, etc.); therefore, it allows an objective assessment of each method. Pairwise retests are conceptually similar to 1-on-1, but they are performed in a low density format using larger volumes of fresh cultures; this explains why interactions missed in the ORFeome-wide 1-on-1 screens still often retest successfully.

Comparison of Y2H methods

We retrieved a total of 156 core PPIs (Supplemental Table 4); 148 of these core PPIs were identified from 1-on-1, STD-1536, or STD-384, the three methods used to screen all 12 baits. Many were recovered by all three methods, but a significant number was also found exclusively by each method (Figure 4A). In particular, although 1-on-1 finds the most PPIs, it misses many PPIs that were found by STD-1536 or STD-384, which indicates that the 1-on-1 screens are not saturating even when repeated twice.

Highly connected baits are known to be challenging for smart-pooling, as they can jeopardize the decoding of results. More precisely, a given smart-pool design can only identify a limited number of positives within a batch, and to increase this number would require a different design with more pools. Consequently, the problem is expected to be less pronounced with STD-1536 than STD-384, which has more preys per batch. Among the 12 baits, eight interact with at most 12 preys (which we grouped as nonhub baits) while the remaining four have between 19 and 35 interactors (grouped as hub baits). This is a high cutoff for defining nonhubs and hubs. It was chosen because the baits used in this study are highly connected overall, and a lower cutoff would have resulted in a nonhubs group with too few data points for a reliable analysis. Using the current cutoff, we estimate our nonhubs category would include the vast majority of proteins in the proteome.

We first compared STD-384 and STD-1536 with 1-on-1 in terms of sensitivity (the percentage of core hits from each method in the whole core data set). When considering nonhub baits, the STD pooling approach performed very well: STD-1536 is as sensitive as 1-on-1, and STD-384 is even significantly more sensitive (61.5% versus 44.2%, *P*-value 0.004 calculated using a binomial distribution; Fig. 4B). This shows that, even when preys are arrayed individually, a significant number of false-negatives occur and cannot be recovered in 1-on-1 analysis. Naturally, false-negative spots are also frequent in STD arrays, but due to the high extra redundancy the STD designs often succeed in coping with them. On the other hand, when considering hub baits, 1-on-1 is the most sensitive followed by STD-1536 and STD-384 (77.9%, 64.4%, and 49.0%, respectively). With hubs, the advantage conferred by the high STD redundancy is expected to be offset by the large number of positives, which can saturate the STD designs such that some interactions cannot be deciphered. Such saturation was evident in the pilot experiment (Supplemental Note) but did not clearly occur with STD-1536 or STD-384, where other factors must have come into play.

In terms of specificity (Fig. 4C), all three methods display very satisfactory Positive Predictive Values (PPVs, i.e., the percentage of

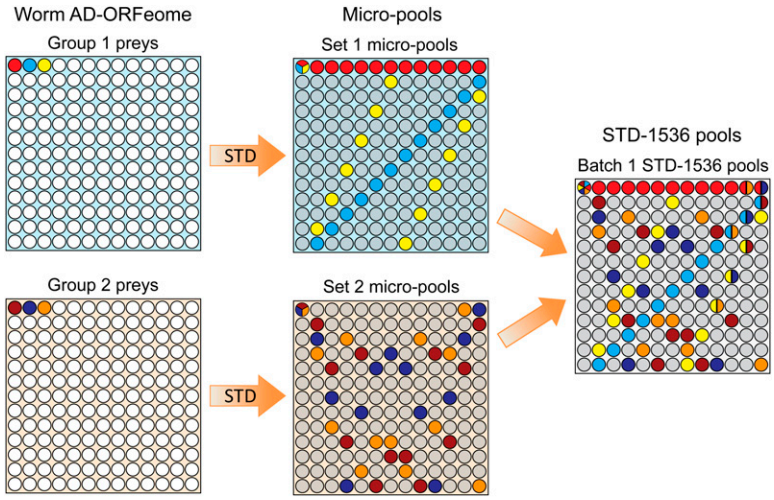

Figure 2. STD pooling design used in this paper. Worm AD-ORFeome preys (12,675) were split into 75 groups, each containing 169 preys. Each group was STD-pooled into a set of 169 micropools. Each micropool contains 13 preys (micropool size: 13), and each prey is contained in a unique combination of 13 micropools (a unique signature; redundancy: 13), as illustrated with three color-coded preys in sets 1 or 2. Two preys co-occur in at most one micropool, so that each prey is uniquely identified by any two of the 13 pools that contain it; therefore, these micropool designs have an extra redundancy of 11. In addition, the micropool signatures of preys with identical AD-ORFeome coordinates from groups 1 and 2 are very different (e.g., light red in set 1 and dark red in set 2): Every two sets of micropools can be superposed to obtain one batch of STD-1536 pools, such that two preys from different groups co-occur in at most two common pools. Consequently, in STD-1536 each prey is uniquely identified by any three of the 13 pools that contain it: STD-1536 pools possess an extra redundancy of 10.

candidate hits found by a given method that passed pairwise retest and ended up in core), averaging at 75%, 78%, and 91% for 1-on-1, STD-384, and STD-1536, respectively. 1-on-1 is the only one to significantly differ between nonhubs and hubs, decreasing from 88.5% to 71.7%, but this change is not surprising because it correlates with the increased 1-on-1 sensitivity.

The higher sensitivity of STD-384 over STD-1536 for nonhubs (Fig. 4B) presumably results from the lower density. For example, weakly positive spots are easier to score in 384 format: this allows identification of genuine Y2H-weak interactions, although it also results in slightly lower PPV for STD-384 (Fig. 4C). Additionally, miniaturization increases the influence of random fluctuations, making it harder to have consistent optimal conditions in 1536 format: Small variations in factors that result in lower signal or higher background have a stronger influence. In particular, in preliminary experiments we noticed that the amount of cells transferred to the target plates is an important parameter. Since the 1536-format pins are 0.7 mm in diameter compared to 1 mm for 384-format pins, they transfer fewer cells and the effect of experimental variability is greater. This observation is not limited to STD pooling but potentially applies to all high density experiments. Indeed, this explains why the two 1536-format assays used here, 1-on-1 and STD-1536, obtain sensitivities that are similar and significantly lower than STD-384 when screening nonhub baits. This is not contradictory with the high repeatability rates that we observed (see below), because our duplicated screens were designed to study the variability intrinsic to each method rather than that due to external parameters: The duplicates were performed in parallel, using the same batches of source and target plates and very similar experimental conditions.

The six baits screened with the Screen-Seq approach include two hub baits and four nonhub baits (Supplemental Table 1). Sensitivities (Fig. 4D) and PPVs (Fig. 4E) were calculated by restricting each data set to the Screen-Seq baits. Screen-Seq displays a high PPV similar to that of the other methods. In terms of sensitivity, Screen-Seq was surpassed by STD-1536 and STD-384 by factors of 1.8 and 1.1 for hubs, and it was surpassed by factors of 2.1 and 2.4 for nonhubs. Furthermore, the selected Screen-Seq baits were among the most connected in both the hubs and nonhubs groups. This explains the reduced sensitivity of STD-384 when restricting it to these six baits (Fig. 4D versus 4B), and biases the comparison in favor of Screen-Seq. Since nearly all baits in a proteome are nonhubs, we estimate that, in a large-scale interactome mapping project where the selected method is applied once or twice, STD would be at least twice as sensitive as Screen-Seq.

We then examined STD-SL, where three hub baits were screened against the STD-1536 pools arrayed in 384-format. STD-SL was not more sensitive than STD-1536 (Supplemental Fig. 3A), indicating that high plate density did not impact STD-1536. Concerning specificity, a single STD-SL candidate failed pairwise retest, entailing an almost perfect PPV (98.2%; Supplemental Fig. 3B). This shows that, with the STD-SL design, the problem of false-positives is virtually eliminated.

Based on our two replicates of each screening, we studied the repeatability of each method for core and FP hits (Supplemental Fig. 4). Core hits are largely repeatable for all methods: The fraction of PPIs identified in both replicates ranges from 65% for STD-1536 up to 86% for 1-on-1 and STD-SL. Concerning FP hits, they are almost never found in both repeats of STD data sets, as expected. However, they are surprisingly repeatable in 1-on-1 and also in Screen-Seq, although this is less significant since there were only seven Screen-Seq false-positives. This may be partly due to localized problems such as cross-contamination in the 1-on-1 master array, which could lead to repeatable false-positives in 1-on-1. Localized problems in the STD arrays would not have such a pronounced effect, because each prey is present in 13 pools that are distributed across the array: STD appears particularly robust with regards to contamination.

Due to its high redundancy, STD can provide valuable information in terms of error rates. False-positive spots were rare in our hands, but false-negatives were frequent, and the false-negative rate appeared largely variable between interactions. Furthermore, interactions yielding a strong signal in one STD series were often strong in other series: The "Y2H strength" of an interaction appears mostly reproducible. However, our data set is too small to draw conclusions on specific baits or preys: Application of highly redundant smart-pooling on a larger scale would be necessary to identify poorly performing baits and preys in a proteome.

Discussion

We have demonstrated that STD-based smart-pooling is a feasible and flexible strategy for mapping PPIs by Y2H at the scale of a complete ORFeome, and we have shown that the method can

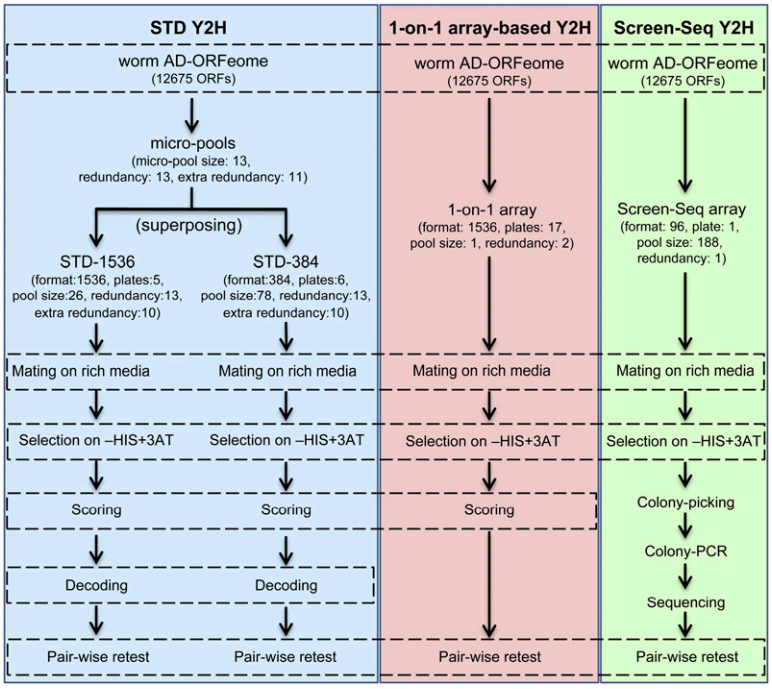

Figure 3. Flowchart of Y2H approaches used in this paper, comparing the main steps of STD Y2H, 1-on-1 array-based Y2H, and Screen-Seq Y2H.

take advantage of high density 384 and 1536 formats. We have compared it with the established Screen-Seq high-throughput method (Rual et al. 2005), and with the labor-intensive "gold standard" one-on-one array-based Y2H (1-on-1) (Uetz et al. 2000).

We separately analyzed "nonhub" baits that were involved in at most 12 interactions, and the more highly connected "hub" baits. This cutoff was chosen because overall the baits used here have many interactions, but from a broader perspective it is quite a high cutoff. For example, only 42 *Saccharomyces cerevisiae* proteins, representing <1% of the proteome, are involved in more than 12 interactions in the "Y2H-Union" data set (Yu et al. 2008), which merges the three proteome-wide *S. cerevisiae* Y2H data sets published to date (Uetz et al. 2000; Ito et al. 2001; Yu et al. 2008). The nonhubs in this study are therefore representative of the vast majority of proteins in an organism and can serve as a useful guide for choosing an approach, while the hubs are informative in that they represent the worst-case scenario for smart-pooling methods.

Every candidate interaction underwent pairwise retest in quadruplicate. This showed that all methods were highly specific in our hands: At least 75% of each method's candidates retested successfully (91% for STD-1536). Screen-Seq is the least sensitive by a factor of up to 2.4 (for nonhubs versus STD-384). When considering nonhub baits, STD-384 and to a lesser extent STD-1536 were very successful: Their sensitivity attains or even exceeds that of 1-on-1, with a 39% increased sensitivity for STD-384 compared to 1-on-1, despite being much more cost- and labor-efficient. This demonstrates the advantage conferred by highly redundant STD pools. As anticipated, STD pooling performed less well with the highly connected baits, yet it remained satisfactory: Sensitivity was intermediate between Screen-Seq and 1-on-1, and as expected due to its smaller batch size, STD-1536 was more resilient to hubs than STD-384. Because the large majority of proteins in *C. elegans* and other organisms are not PPI hubs (Barabasi and Oltvai 2004; Gandhi et al. 2006), and because of the previously discussed large cutoff value used for defining nonhubs in this study, STD could be very useful as a highly sensitive and efficient first pass for large-scale interactome mapping. Any exceptionally strong hubs could be subsequently screened more deeply using another method such as 1-on-1, or by sequencing positive colonies that cannot be decoded unambiguously in the STD screen.

STD-1536 and STD-384 require five and six plates, respectively, while Screen-Seq fits on a single plate and 1-on-1 needs 17 plates. We have shown that, due to its high redundancy, the STD method is not affected by de novo autoactivators, which arise by acquiring mutations during the screening process, and the Screen-Seq step of cycloheximide counter-selection (Rual et al. 2005) can be safely skipped. Additionally, positives are directly identified in STD pooling, whereas Screen-Seq resorts to colony picking and sequencing (Fig. 3). Altogether, we estimate that the STD workload and costs are approximately three times higher than those of Screen-Seq, while coverage is increased at least twofold. In contrast, performing three repeats of Screen-Seq only improves coverage by 30% relative to single-pass Screen-Seq (Yu et al. 2008).

Comparing now with 1-on-1, since the Y2H screening steps are identical, the STD approach is approximately three times more cost- and labor-efficient, while being in fact more sensitive except for the few strong hub baits. STD also requires an initial investment to build the STD pools, but this is a one-time expenditure, as the pools can be copied and used many times. In addition, we designed and built intermediate micropools, which can be simply superposed to generate larger STD pools of various sizes, such as STD-1536 and STD-384 used here. This strategy minimizes the costs of building STD pools and provides greatly increased flexibility: The complex cherry-picking step for building micropools is performed a single time, and building STD pools of diverse pool and batch sizes is then a quick and cheap procedure, allowing adaptation of the pooling design to specific assay conditions.

Two other smart-pooling methods have been recently used to map PPIs (Jin et al. 2006, 2007) and protein–DNA interactions (Vermeirssen et al. 2007). However, they relied on designs that lack flexibility and possess an extra redundancy of at most one. This limits their ability to deal with the high false-positive and false-negative rates that are common in many assays, so that identifying the positives in these studies required sequencing positive colonies or retesting many ambiguous candidates. In contrast, STD is very flexible and one can choose a high extra redundancy if desired, for example 10 as used in this study. This allows us to successfully deal with high levels of noise, without any need for sequencing and without generating large numbers of low-confidence candidates, as shown by the high PPV values obtained with our STD designs.

In summary, we showed the application of the STD pooling strategy in ORFeome-wide Y2H screening and compared it with

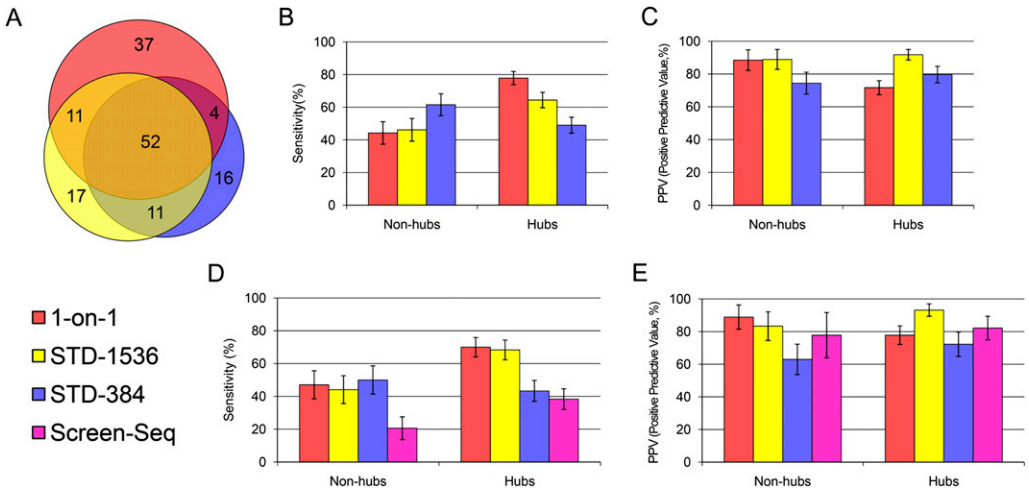

Figure 4. Comparison of Y2H results. (*A*) Area-proportional Venn diagram of PPIs found by 1-on-1, STD-1536, and STD-384 (generated by http://venndiagram.tk/). (*B*) Sensitivity (the percentage of core hits from each method in the whole core set) and (*C*) positive predictive value (PPV; the percentage of each method's hits that successfully pass pairwise retest and end up in core) of 1-on-1, STD-1536, and STD-384, restricted either to hub baits or to nonhub baits. (*D*) Sensitivity and (*E*) PPV of 1-on-1, STD-1536, STD-384, and Screen-Seq, restricting all data sets to the four nonhub and two hub baits screened in Screen-Seq. Error bars indicate standard error.

established high-throughput approaches, one-on-one array-based Y2H (Uetz et al. 2000) and Screen-Seq (Rual et al. 2005). Screen-Seq remains an appropriate method for quickly producing low-coverage interactomes, while STD appears as the method of choice for obtaining maps with higher coverage. STD pooling is also more powerful and flexible than other recently employed pooling designs (Jin et al. 2006, 2007; Vermeirssen et al. 2007). We expect that STD-based smart-pooling can be applied in other large-scale functional genomics experiments that rely on a basic yes-or-no test to identify rare positive events, provided that pools can be tested and yield a positive signal if they contain at least one positive, such as yeast one-hybrid, drug screening (e.g., Kainkaryam and Woolf 2008), or PCR- or hybridization-based analyses (e.g., Wu et al. 2008).

Methods

Details on the STD designs

In theory, with a redundancy of 13 and a design comprising 169 pools per batch (as we used in our STD pooling), STD can make pools for up to 13^{13} preys per batch, although the pool size increases proportionately with the number of preys per batch. Going down from 13^{13}, the extra redundancy starts at zero and increases by one each time the exponent decreases. For example, between 14 (13^1 + 1) and 169 (13^2) preys per batch, any two preys co-occur in at most one common pool (leaving an extra redundancy of 11, as in the worm micropools); while between 170 (13^2 + 1) and 2197 (13^3) preys per batch, a pair of preys co-occurs in at most two pools (leaving an extra redundancy of 10, as in STD-1536 or STD-384).

Every 12 sets of worm micropools (169 preys per set) is a collection of subdesigns of an STD design with 2028 preys per batch (whose extra redundancy is 10). They can therefore be superposed to obtain the original STD design. Each individual set of micropools is also isomorphic to a smaller STD design, and can be used as an STD pooling batch in its own right, with an extra redundancy of 11. When two or six consecutive micropool sets are superposed to obtain STD-1536 or STD-384, the resulting designs are again isomorphic to an STD design with 228 or 1014 preys per batch, respectively, and therefore they both have an extra redundancy of 10. More specifically, worm micropools are subdesigns of STD(2028;13;13) isomorphic to STD(169;13;13), and were superposed to obtain designs isomorphic to STD(338;13;13) for STD-1536 and STD(1014;13;13) for STD-384 (see Thierry-Mieg 2006a for details).

Building STD pools

The sources were all Worm ORFeome v1.1 and v3.1 AD plates (11001 to 11114 and 31001 to 31022). The source plates were thawed at room temperature, inoculated in 96-format deep well plates containing SD-Trp liquid media, and incubated at 30°C for 2 d. After resuspension by shaking, micropools were assembled in 96-format deep well plates by cherry-picking using a Tecan Freedom EVO liquid handling robot (Tecan Group Ltd.). The robot was programmed directly in GWL (Supplemental Data), which optimized the process. STD-1536 and STD-384 pools were generated in 384-format and 96-format (STD-384 only) by superposing the appropriate micropool plates with a Tecan Aquarius MultiChannel Pipetting robot (Tecan Group Ltd.). All pools were frozen and stored at −80°C with 20% glycerol.

Handling 1-on-1 and STD arrays

Before Y2H screening, 1-on-1 or STD arrays glycerol stock plates were thawed at room temperature, mixed thoroughly with a plate shaker and transferred to SD-Trp agar plates with a "BM3-SC+Carousel" robot (S&P Robotics). After incubation, this set of "master" agar plates was replicated into multiple copies (up to eight), which could each be used either for screening or as a source for further replications. However, fresh arrays should still be occasionally remade from glycerol stock, because the STD arrays begin losing representation after more than five sequential replications (data not shown). The arrays appear fully functional after being stored at 4°C for at least 2 mo, although in this study we used them within 1 wk after replication to avoid confounding factors and guarantee the highest data quality.

Y2H screening with 1-on-1 and STD arrays

The Y2H screening with 1-on-1 and STD arrays was performed using a "BM3-SC+Carousel" robot (S&P Robotics) following the

previously reported protocol (Uetz et al. 2000), except that the diploid selection step on SD-Leu-Trp was skipped: Preliminary experiments showed that, in our hands, including this step did not result in any improvements, perhaps because the robotic replication step may not be fully effective in transferring all components of a colony spot, so that the additional replication step compensates any gains from the diploid selection. We used pins of 1 mm diameter for 384-format and 0.7 mm for 1536-format. 1-on-1 spots were scored manually using the in-house ColonyImager image-processing program (H Ding and C Boone, unpubl.) as positive or negative. Each prey is present in duplicate on the 1-on-1 arrays; a 1-on-1 hit obtained a confidence score of Weak if it was positive in a single spot and Strong if it was positive in both duplicate spots. STD spots were scored similarly, except we used four discrete levels for each spot: strong (clear positive) or weak (smaller than strong but well above background) for positives, and none (no detectable signal) or faint (barely above background, most likely negative) for negatives. These results were transformed into a suitable XML format with Perl scripts, and decoded with interpool (Thierry-Mieg and Bailly 2008). The "distance" parameter δ was chosen to fit our experimental conditions. It turned out that false-positives were relatively rare while false-negatives were common, leading us to use a very sensitive distance ($\delta_{NONE} = 2$, $\delta_{FAINT} = 1$, $\delta_{WEAK} = 4$, $\delta_{STRONG} = 6$). Clearly this choice did not strongly compromise specificity, as shown by the STD PPV values obtained after pairwise retesting (Fig. 4C). All relevant scripts, programs, and data files are available (Supplemental Data). A confidence score was attributed to each STD hit, depending solely on the number of putative false-negative spots for the hit. Specifically, "none" spots carry a cost of 2 and "faint" spots 1, and summing over all false-negatives for a hit yields a total cost; if this total cost is at most 4 the confidence score is 5, if it is up to 8 the score is 4, and so on until reaching the lowest confidence score of 1 if the total cost is between 17 and 20. All results were imported into a custom database for further analysis.

Y2H screening with the Screen-Seq approach

The 188 preys from every two worm ORFeome plates were pooled together. All resulting pools were assembled into one 96-well plate to generate the so-called "superpool" plate. Each bait was screened against the superpool plate using the method reported before (Rual et al. 2005). At most three positive colonies were picked from each spot, and prey inserts were amplified by colony PCR and sequenced for identification.

Pairwise retest

Retests were performed in quadruplicate by scoring a single phenotype of the *HIS* reporter in 96-format on agar plates, using 5 μL of bait and 5 μL of prey fresh cultures from archival stocks. Each retest was scored as negative, weak, or strong (0, 1, or 2, respectively). Summing over the four replicates, we obtained a retest score between 0 and 8 for each hit. Core hits are those whose retest score was at least 6, while hits with scores at most 2 were classified as FP and the remaining hits with intermediate retest scores were classified as noncore.

Author contributions

X.X., D.E.H., M.V., C.B., and N.T.M. conceived the project. X.X. and N.T.M. designed the experiments. J.F.R., T.H.K., and N.T.M. performed the pilot project experiments. N.T.M. designed the STD arrays, and X.X., D.E.H., and N.T.M. built the STD arrays. X.X. performed the ORFeome-wide Y2H experiments and scored the plates. X.X. and N.T.M. performed the computational analyses, produced the figures, and wrote the manuscript.

Acknowledgments

We thank Haiyuan Yu, Jingjing Li, and Olivier Francois for help with the statistical analysis. This work was supported by a Canadian Cancer Society grant awarded to C.B., and grant R01-HG001715 from the National Human Genome Research Institute of the National Institutes of Health awarded to M.V. M.V. is a "Chercheur Qualifié Honoraire" from the "Fonds de la Recherche Scientifique" (FRS-FNRS, French Community of Belgium).

References

Barabasi AL, Oltvai ZN. 2004. Network biology: Understanding the cell's functional organization. *Nat Rev Genet* **5:** 101–113.

Bruno WJ, Knill E, Balding DJ, Bruce DC, Doggett NA, Sawhill WW, Stallings RL, Whittaker CC, Torney DC. 1995. Efficient pooling designs for library screening. *Genomics* **26:** 21–30.

Deplancke B, Mukhopadhyay A, Ao W, Elewa AM, Grove CA, Martinez NJ, Sequerra R, Doucette-Stamm L, Reece-Hoyes JS, Hope IA, et al. 2006. A gene-centered *C. elegans* protein-DNA interaction network. *Cell* **125:** 1193–1205.

Gandhi TK, Zhong J, Mathivanan S, Karthick L, Chandrika KN, Mohan SS, Sharma S, Pinkert S, Nagaraju S, Periaswamy B, et al. 2006. Analysis of the human protein interactome and comparison with yeast, worm and fly interaction data sets. *Nat Genet* **38:** 285–293.

Ito T, Chiba T, Ozawa R, Yoshida M, Hattori M, Sakaki Y. 2001. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc Natl Acad Sci* **98:** 4569–4574.

Jin F, Hazbun T, Michaud GA, Salcius M, Predki PF, Fields S, Huang J. 2006. A pooling-deconvolution strategy for biological network elucidation. *Nat Methods* **3:** 183–189.

Jin F, Avramova L, Huang J, Hazbun T. 2007. A yeast two-hybrid smart-pool-array system for protein-interaction mapping. *Nat Methods* **4:** 405–407.

Kainkaryam RM, Woolf PJ. 2008. poolHiTS: A shifted transversal design based pooling strategy for high-throughput drug screening. *BMC Bioinformatics* **9:** 256. doi: 10.1186/1471-2105-9-256.

Lamesch P, Milstein S, Hao T, Rosenberg J, Li N, Sequerra R, Bosak S, Doucette-Stamm L, Vandenhaute J, Hill DE, et al. 2004. *C. elegans* ORFeome version 3.1: Increasing the coverage of ORFeome resources with improved gene predictions. *Genome Res* **14:** 2064–2069.

Reboul J, Vaglio P, Rual JF, Lamesch P, Martinez M, Armstrong CM, Li S, Jacotot L, Bertin N, Janky R, et al. 2003. *C. elegans* ORFeome version 1.1: Experimental verification of the genome annotation and resource for proteome-scale protein expression. *Nat Genet* **34:** 35–41.

Ren R, Mayer BJ, Cicchetti P, Baltimore D. 1993. Identification of a ten-amino acid proline-rich SH3 binding site. *Science* **259:** 1157–1161.

Rual JF, Venkatesan K, Hao T, Hirozane-Kishikawa T, Dricot A, Li N, Berriz GF, Gibbons FD, Dreze M, Ayivi-Guedehoussou N, et al. 2005. Towards a proteome-scale map of the human protein–protein interaction network. *Nature* **437:** 1173–1178.

Stelzl U, Worm U, Lalowski M, Haenig C, Brembeck FH, Goehler H, Stroedicke M, Zenkner M, Schoenherr A, Koeppen S, et al. 2005. A human protein–protein interaction network: A resource for annotating the proteome. *Cell* **122:** 957–968.

Thierry-Mieg N. 2006a. A new pooling strategy for high-throughput screening: The shifted transversal design. *BMC Bioinformatics* **7:** 28. doi: 10.1186/1471-2105-7-28.

Thierry-Mieg N. 2006b. Pooling in systems biology becomes smart. *Nat Methods* **3:** 161–162.

Thierry-Mieg N, Bailly G. 2008. Interpool: Interpreting smart-pooling results. *Bioinformatics* **24:** 696–703.

Tong AH, Drees B, Nardelli G, Bader GD, Brannetti B, Castagnoli L, Evangelista M, Ferracuti S, Nelson B, Paoluzi S, et al. 2002. A combined experimental and computational strategy to define protein interaction networks for peptide recognition modules. *Science* **295:** 321–324.

Tong AH, Lesage G, Bader GD, Ding H, Xu H, Xin X, Young J, Berriz GF, Brost RL, Chang M, et al. 2004. Global mapping of the yeast genetic interaction network. *Science* **303:** 808–813.

Uetz P, Giot L, Cagney G, Mansfield TA, Judson RS, Knight JR, Lockshon D, Narayan V, Srinivasan M, Pochart P, et al. 2000. A comprehensive analysis of protein–protein interactions in *Saccharomyces cerevisiae*. *Nature* **403:** 623–627.

Venkatesan K, Rual JF, Vazquez A, Stelzl U, Lemmens I, Hirozane-Kishikawa T, Hao T, Zenkner M, Xin X, Goh KI, et al. 2009. An empirical framework for binary interactome mapping. *Nat Methods* **6:** 83–90.

Vermeirssen V, Deplancke B, Barrasa MI, Reece-Hoyes JS, Arda HE, Grove CA, Martinez NJ, Sequerra R, Doucette-Stamm L, Brent MR, et al. 2007. Matrix and Steiner-triple-system smart pooling assays for high-performance transcription regulatory network mapping. *Nat Methods* **4:** 659–664.

Wu Y, Liu L, Close TJ, Lonardi S. 2008. Deconvoluting BAC-gene relationships using a physical map. *J Bioinform Comput Biol* **6:** 603–622.

Yu H, Braun P, Yildirim MA, Lemmens I, Venkatesan K, Sahalie J, Hirozane-Kishikawa T, Gebreab F, Li N, Simonis N, et al. 2008. High-quality binary protein interaction map of the yeast interactome network. *Science* **322:** 104–110.

Zhong J, Zhang H, Stanyon CA, Tromp G, Finley RL Jr. 2003. A strategy for constructing large protein interaction maps using the yeast two-hybrid system: Regulated expression arrays and two-phase mating. *Genome Res* **13:** 2691–2699.

Received December 12, 2008; accepted in revised form April 14, 2009.