Analyse des motifs régulateurs d'expression dans la famille multigénique de gènes WAKs chez le peuplier

Moufid Mejdoub

Master II Biologie et Environnement

Spécialité

Génomique, Écophysiologie et Production Végétales (GEPV)

Analyse des motifs régulateurs d'expression dans la famille multigénique de gènes WAKs chez le peuplier

Rapport de stage rédigé par :

Moufid Mejdoub

Responsable du stage :

Patricial Roeckel-Drevet, Philippe Label, Jean-Stéphane Venisse

UMR PIAF 547

Physique et Physiologie Intégratives

de l'Arbre Fruitier et Forestiers

Les Cézeaux, 24 Avenue des Landais

63177 Aubière

Juin 2016

Remerciements :

Je tiens tout d'abord à remercier Jean-Louis JULIEN de m'avoir accueilli au sein de l'UMR PIAF afin d'effectuer mon stage recherche de Master 2ème année

Je souhaite également remercier mes encadrants de stage, Patricia ROECKEL-DREVET, Philippe LABEL et Jean Stéphane VENISSE pour leurs conseils et le soutient apportés tout au long du stage, ainsi que sur l'écriture de ce rapport.

Je souhaite enfin remercier tous le personnel du PIAF ainsi qu'à Kévin TOCQUARD, Marie GARAVILLON, Pierrick BENOÎT et Mélyne FALCON pour leurs aides, leurs soutient, et la bonne humeur qu'ils apportent au quotidien.

Résumé

Les protéines WAKs (pour Wall-Associated Kinases) ont une structure transmembranaire avec une partie extracellulaire liée à la paroi et une partie kinase intracellulaire. Cette caractéristique structurelle fait d'elles des candidates pour intervenir dans la transduction de signaux pariétaux. Chez *Populus*, les protéines WAKs sont codées par une famille multigénique de 175 membres. Ces gènes sont différentiellement exprimés en réponse à différentes conditions comme l'attaque d'un pathogène, l'inclinaison et les carences minérales. Afin de progresser dans la compréhension des rôles des différents membres de cette famille, nous avons initié une étude de régions régulatrices de la transcription au niveau du promoteur (1000 pb) et du premier intron. Les outils bio-informatiques utilisés ont permis d'analyser ces séquences régulatrices, et de réaliser une corrélation entre les motifs retrouvés et l'expression des WAKs en réponse à différentes contraintes environnementales. Cent douze motifs sont représentés significativement dans ces 2 régions régulatrices. Une corrélation a été mise en évidence entre le motif RGWCGTG et une stimulation gravitropique. De même, le motif AWTTCAAA semblerait être lié au changement d'état de turgescence foliaire. Ces résultats préliminaires permettent d'avancer dans la connaissance de cette large famille multigénique chez *Populus trichocarpa.*

Mots clefs : Wall associated kinase, motifs, séquences régulatrices, promoteur, premier intron, ACP

The WAKs proteins (for Wall-Associated Kinases) present a transmembrane structure with extracellular part linked ti wall components and an intracellular kinase portion. This structural characteristic makes them interesting candidates tointerviene intervening in the transduction of wall signals. In Populus, the WAKs proteins are encoded by a multigenic familly of 175 members. These genes are differentially expressed in response to several conditions like pathogen attacks, tilting or mineral deficiency. In order to progress in the understanding of the roles of the different familly members, we started a study of regulatory regions of the transcription in the promoter (1000 pb) and the first intron. The bio-informatics tools used allowed to analyse these analysis of these regulatory sequences, and to look for a correlations between the motifs found and WAKs expression in response to different environmental constraints. The results showed that 112 motifs are significantly represented in these two regulatory regions. A correlation has been highlighted between the motif RGWCGTG and gravitropic stimulation. Similarly, the motif AWTTCAAA seems to be related to the variation of leaf turgor state. These preliminary results allowed to progress in knowledge of this large multigenic family in *Populus trichocarpa*

Key words : Wall Associated Kinase, motifs, regulatory region, promoter, first intron, ACP

Liste des figures et tableaux

Table 5 :Extrait des résultats obtenus avec l'outil 'analyseMotifs'

Table 6 : Extrait des résultats obtenus avec l'outil 'testMotifs'

Liste des abréviations

ACP :Analyse en Composante Principale

ADNc :Acide DésoxyriboNucléique complémentaire

ARN :Acide RiboNucléique

ANR :Agence Nationale de Recherche

BRE :B-Recognition Elements

DPE :Downstream Promoteur Element

AtGRP-3 :Glycine-Rich secreted cell-wall Proteins-3

Inr :Initiator

MTE :Motif Ten Element

PIAF :Physique et Physiologie Intégrative de l'Arbre Fruitier et Forestier

PIC :PreInitiation Complex

PR :Pathogen Related

qPCR :quantitative Polymerase Chain Reaction

SA:Salicylic Acid

TSS :Transcription Start Site

UAS :Upstream Activating Sequence

URS :Upstream Repressing Sequence

WAK :Wall Associated Kinase

Sommaire

1.Introduction

L'Unité Mixte de Recherche PIAF (Physique et Physiologie Intégratives de l'Arbre Fruitier et Forestier) est un laboratoire composé de 3 équipes étudiant l'impact du changement climatique sur le développement et le fonctionnement des arbres. Chaque équipe a un champ d'étude particulier. Ainsi l'équipe « Hydro » (Hydrolique et résistance à la sécheresse des arbres) traite des questions de physiologie hydrique d'une plante, l'équipe « MEA » (Micro Environnement et Arbre) s'attarde à mieux comprendre l'arbre et son environnement thermique, lumineux ou minéral, et l'équipe « MECA » (Contraintes Mécaniques et activités des zones en croissance) étudie l'impact de contraintes mécaniques sur la plante (vent, gravité….). Ces travaux sont menés à différentes échelles, allant de l'ADN jusqu'à la plante entière. Le principal modèle végétal utilisé est le peuplier et cela pour 3 raisons : son génome est entièrement séquencé (troisième génération d'annotation), c'est une espèce ligneuse qui peut être cultivée et transformée *in vitro* et qui a un intérêt pour la filière bois.

Des recherches menées à l'UMR PIAF portent sur une famille de protéines à la structuration particulière. Les protéines WAKs pour Wall Associated Kinases, ont pour la majorité une structure transmembranaire présentant une région extracellulaire qui permet potentiellement une liaison avec les pectines de la paroi, et un domaine kinase intracellulaire. D'autres membres de la famille des WAKs, ne possèdent que la région extracellulaire ou la région cytoplasmique. En revanche, la majorité de la famille des WAKs ont les caractéristiques qui désignent ces protéines comme des candidats pour jouer le rôle de récepteur de signaux pariétaux à l'initiation d'une signalisation cellulaire.

Des études sur les WAKs ont été réalisées chez *Arabidopsis thaliana* et chez *Oryza sativa* et sont présentées dans la synthèse bibliographique. Récemment Tocquard et al. (2014) décrivent *in silico* la structure de la famille multigénique chez *Populus trichocarpa.*

Des motifs régulateurs spécifiques dans la séquence promotrice de quelques WAKs ont été recherchés chez *A. thaliana* (Meier et al., 2010) et chez *O. sativa* (Hu et al., 2014). A notre connaissance, aucune étude n'a porté sur les promoteurs des gènes WAKs chez *P. trichocarpa*.

Mon stage de recherche de seconde année de master, en coopération entre les équipes « MECA » et « Hydro », a pour objectif essentiel d'étudier les séquences régulatrices des gènes WAKs. L'objectif de mon stage est :

- (i) d'identifier des motifs dans le promoteur et 1er intron des gènes *Pt*WAKs qui sont fortement représentés dans cette famille par rapport au reste du génome.

-(ii) de vérifier s'il existe une relation entre la présence de motifs particuliers et l'expression des WAKs à différentes contraintes.

La synthèse bibliographique qui suit, s'attachera à présenter des concepts et points clefs du contexte tels que :

1. La présentation de la protéine (WAK), des différents domaines qui la constituent et de ses différents rôles physiologiques connus à ce jour.
2. L'état des connaissances actuelles sur les séquences régulatrices au niveau du promoteur et du 1er intron.

2.Bibliographie

2.1 Les WAKs

Les WAKs (Wall Associated Kinases) sont des récepteurs transmembranaires localisés au niveau de la membrane plasmique des cellules. Ces protéines sont codées par une famille multigénique, bien étudiée chez le riz et chez Arabidopsis (He et al., 1999 ; Verica et al., 2003 ; Verica et He, 2002). Ces récepteurs membranaires ont la particularité de se lier aux parois des cellules végétales (Decreux et Messiaen, 2005 ; He et al., 1996) et plus précisément aux pectines qui la composent. Ils interviennent dans différents rôles physiologiques, comme la croissance cellulaire (Kohorn et al., 2006) ou la défense contre les pathogènes (Delteil et al., 2016 ; He et al., 1998 ; Kohorn et al., 2009 ; Kohorn et al., 2012b).

2.1.1 Description de la famille multigénique

La famille des WAKs est relativement peu connue. Elle a été caractérisée chez *Arabidopsis thaliana* (He et al., 1999 ; Verica et al., 2003 ; Verica et He, 2002), chez *Oryza sativa* (Zhang et al., 2005) et *Populus trichocarpa* (Tocquard et al., 2014). On commence tout juste à la décrire chez *Hordeum vulgare* (l'orge) (Kaur et al., 2013).

Chez Arabidopsis, 26 gènes ont été identifiés ; 5 membres *AtWAK* (He et al., 1999) regroupés dans une portion de 30 kb sur le chromosome 1 et 21 autres *AtWAKs-like* (*AtWAKLs*) répartis sur l'ensemble des 5 chromosomes (Verica et He, 2002).

Les gènes *AtWAKs* codent des protéines composées d'une partie N-terminale apoplastique, d'une unique région transmembranaire et d'une partie C-terminale cytosolique (He et al., 1999). La région C-terminale possède un domaine Sérine/Thréonine kinase qui est conservé à 86% entre chacun des membres. La séquence protéique des parties extracellulaires des 5 AtWAKs présente une similarité de 40 à 64%. Chaque protéine possède 2 domaines EGF-like (Epidermal Growth Factor) adjacents à la région transmembranaire. Chez Arabidopsis, deux types de domaines EGF-like ont été rapportés : EGF-2 et EGF-Ca^{2+} (Kohorn, 2001). Chez les animaux, les domaines EGF participent à la dimérisation de protéines dans des conformations induites par le calcium (Stenflo et al., 2000).

Des analyses bio-informatiques ont été réalisées en prenant le gène *AtWAK1* comme référence ce qui a permis l'identification de 21 gènes supplémentaires appelés *AtWAKs-like* (*AtWAKL*). Ces gènes sont également répartis en petits clusters sur les 5 chromosomes (Verica et He, 2002). Dix sept *AtWAKLs* sont prédites pour coder des protéines avec une structure transmembranaire comparable aux 5 AtWAKs. Sur la base d'une étude phylogénétique des séquences protéiques d'*At*WAKs et *At*WAKLs, les auteurs distinguent 4 clusters. Un schéma de la structure protéique

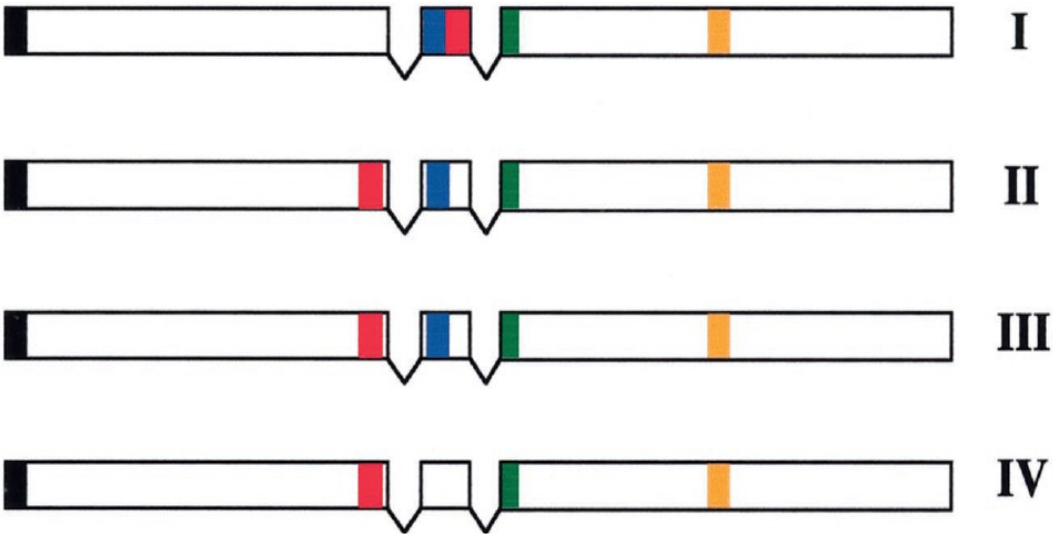

Figure 1. Structure exons/introns des WAKs et WAKLs chez Arabidopsis thaliana d'après Verica et He (2002). Les introns sont représentés par 'V'. Les clusters déterminés d'après une analyse phylogénétique sont notés I à IV. Chaque région codante est indiquée par une boite colorée: la séquence signal N-terminale (noir), le domaine EGF-like (rouge), le domaine EGFCa²+ (bleu), le domaine transmembranaire (vert) et le domaine kinase (jaune).

Type de gènes	Domaine présents	Nombre de gènes
OsWAK-RLK	Contient le domaine EGF-like extracellulaire et le domaine kinase Cytoplasmique	67
OsWAK-RLCK	domaine kinase cytoplasmique Avec une identité ≥40% aux *Os*WAK-RLK	28
OsWAK-RLP	Domaine EGF-like extracellulaire Uniquement	13
OsWAK gène court	<300 acide aminée mais avec ≥40% de la longueur des protéine OsWAK	12
OsWAK pseudogène	codon stop dans le cadre de lecture la région codante	5

Table 1: Classification des OsWAKs en fonction de leur structure protéique d'après Zhang et al., (2005). RLK : Recepteur-Like Kinase, RLCK : Recepteur-Like Cytoplasmic Kinase, RLP: Recepteur-Like Protein.

est réalisé *in silico* pour chaque groupe (Figure 1) (Verica et He, 2002). Le groupe I contient les 5 membres AtWAKs (He et al., 1999) possédant des domaines EGF-2 et EGFCa^{2+} codés par le second exon. Le groupe II est composé des AtWAKLs 1 à 6 et AtWAKL22. Dans ce groupe, les protéines contiennent un domaine EGF-2 et EGF-Ca^{2+}, la séquence du gène d'EGF-2 étant séparée de la séquence d'EGF-Ca^{2+} par le premier intron. Le groupe III regroupe les AtWAKL9, AtWAKL10, AtWAKL11, AtWAKL13, AtWAKL17 et AtWAKL18. Ce groupe est, d'un point de vue structurel, proche du groupe II. Enfin le groupe IV contient les AtWAKL14, AtWAKL15, AtWAKL20 et AtWAKL21. Chez ces 4 membres le domaine EGF-Ca^{2+} est absent alors que le domaine EGF-2 est présent sur l'exon 1. Dans l'ensemble, les AtWAKs et AtWAKLs présentent une structure génomique intron/exon bien conservée et une divergence au niveau de la partie extracellulaire avec parfois les domaines de type EGF altérés. Les autres gènes semblent coder des protéines tronquées correspondant seulement à la partie extracellulaire (Verica et He, 2002).

Le génome du riz (*Oryza sativa* subsp. *japonica*) présente une famille multigénique bien plus importante qu' *Arabidopsis thaliana*. En effet, il contient 125 gènes *WAKs*, classés selon la structure de la protéine (Zhang et al., 2005) (Table 1): 67 sont des OsWAK-RLKs (Receptor-like kinases) contenant à la fois le domaine kinase intracellulaire et les domaines EGF-like extracellulaires, 28 sont des OsWAK-RLCKs (Receptor-like cytoplasmic kinases) contenant uniquement le domaine kinase cytoplasmique présentant au moins 40% d'homologie avec le domaine kinase des OsWAK-RLKs, 13 sont des OsWAK-RLPs (Receptor-like protein) contenant exclusivement un domaine EGF extracellulaire, 12 OsWAKs sont dits « short genes », sans domaine particulier mais présentent au moins 40% d'homologie avec les OsWAKs transmembranaires. Enfin, 5 séquences sont des pseudogènes. Les 125 OsWAKs sont répartis sur l'ensemble des 12 chromosomes du riz. Plus récemment, une analyse faite à partir d'une nouvelle génération d'annotation du génome du riz a dénombré 130 OsWAKs dans le génome de la sous-espèce *japonica* et 111 chez *indica*. Cette différence peut s'expliquer par l'évolution distincte de ces deux sous-espèces (De Oliveira et al., 2014).

2.1.2 La structure des WAKs chez le peuplier

Chez *Populus trichocarpa* 175 membres de WAKs ont été identifiés (Tocquard et al., 2014) : 141 gènes PtWAKs ne sont pas retrouvés dans un unique cluster comme chez *A. thaliana* mais peuvent être regroupés par 3 jusqu'à 10 gènes 15 des 19 chromosomes qui composent le génome de *P. trichocarpa*. Les 34 autres gènes PtWAKs, ne sont pas encore localisés dans le génome de *P. trichocarpa.* En suivant la nomenclature établie par Zhang et al. (2005), les PtWAKs peuvent être

Group	Specifications	Gene number
*Pt*WAK-receptor-like kinase (RLK)	One intracellular kinase domain and one or more extracellular domains (GubWAK, WAK, WAKassoc, and/or EGFs)	119
*Pt*WAK-receptor-like cytoplasmic kinase (RLCK)	Only an intracellular kinase domain	32
*Pt*WAK-receptor-like protein (RLP)	At least one extracellular domains (GubWAK, WAK, WAKassoc, and/or EGF)	22
*Pt*WAK short genes	No domain	2
Total number of WAKs		175

Table 1: Classement des PtWAKs sur la base de leurs domaines protéiques (Tocquard et al., 2014)

classés en 4 groupes (Table 1) : 119, 32, et 22 gènes *Pt*WAKs codent respectivement des RLK, RLCK et RLP, deux gènes sont classés « gènes courts ».

La partie extracellulaire des PtWAKs (RLK et RLP) présente de nombreux domaines (GubWAK, WAK, WAKassoc, EGF-Ca2+, cEGF et EGF3). En commençant par la partie N-terminale, la majorité des protéines PtWAKs possède un domaine GubWAK (wall-associated receptor kinase homogalacturonan-binding), décrit chez Arabidopsis (Decreux et al., 2006). Le domaine GubWAK aurait un rôle de liaison aux composants de la paroi. Chez Arabidopsis, ce domaine se lie aux pectines présentes dans la paroi cellulaire (Decreux et Messiaen, 2005). Ensuite on trouve deux domaines qui n'avaient jamais été caractérisés dans la littérature : le domaine WAK et le domaine WAKassoc (wall associated receptor kinase C-terminal domain). A l'heure actuelle, la fonction des domaines WAK et WAK_assoc reste encore inconnue. Quant à WAK_assoc, il présente une richesse en cystéine (PFAM) (Finn et al., 2016). Enfin, la partie extracellulaire peut présenter trois domaines « EGF-like », qui ne sont pas systématiquement retrouvés ensemble. Le domaine EGF-Ca2+ (calcium-binding EGF domain) est le seul domaine présent à la fois chez Arabidopsis, le riz et le peuplier. Le domaine EGF2 a été décrit chez Arabidopsis et le riz mais pas chez le peuplier. Chez le peuplier, on trouve deux autres domaines EGF-like : cEGF (Clr-like EGF_like domain) et EGF3. Les domaines EGF ont été beaucoup étudiés chez les animaux (Stenflo et al., 2000). Ils sont connus pour interagir dans une conformation induite par le calcium. La présence d'au moins un domaine EGF-like suggère la possibilité d'interactions protéine/protéine au niveau de la partie extracellulaire des WAKs.

Une analyse phylogénétique portant sur la partie kinase des WAKs a montré chez *Populus trichocarpa*, *Oryza sativa* et *Arabidopsis thaliana* que la famille se compose de deux clades. Dans le clade I on trouve des membres provenant de 3 espèces, supposant l'existence d'un ancêtre commun remontant à la séparation entre les monocotylédones et les dicotylédones.
Dans le clade II, on ne trouve que des membres WAKs de peuplier suggérant une évolution indépendante de ce clade au sein de *Populus* (Tocquard et al., 2014).

2.1.3 La capacité d'interaction des WAKs

He et al. (1996) ont démontré qu'il existe une liaison entre AtWAK1 et la paroi via leur domaine extracellulaire. L'utilisation d'anticorps anti-AtWAK1 a permis de localiser la protéine au niveau du continuum paroi – membrane plasmique, et à l'aide de divers protocoles utilisant une variété d'enzymes et de détergents, les protéines AtWAK1 peuvent être isolées (He et al., 1996).

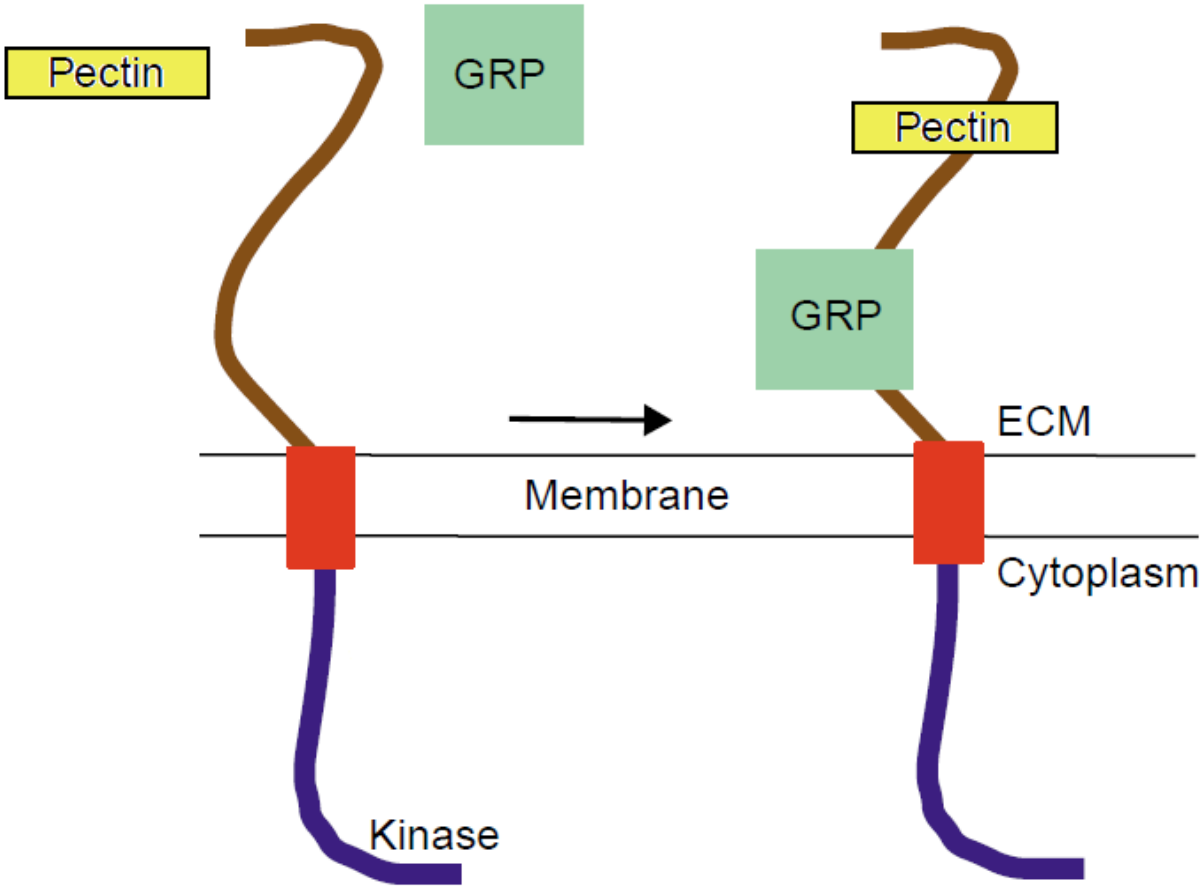

Figure 2: La capacité de liaison de la partie extracellulaire des WAKs avec la pectine et des GRPs (Glycine-Rich Protein). ECM (Extra Cellular Matrice) (Kohorn, 2001)

2.1.3.1 Interaction avec des sucres pariétaux

La seule utilisation d'une pectinase par Wagner et Kohorn (2001) a permis d'extraire AtWAK1, indiquant que la liaison s'établit entre les WAKs et les pectines de la paroi (Figure 2). Ceci a été confirmé en utilisant un anticorps anti-WAK1 et des anticorps anti-pectine (JIM5 et JIM7), proposant aussi l'existence d'une liaison covalente entre les deux (Wagner et Kohorn, 2001). Les WAKs semblent sensibles à l'état des pectines ; lorsqu'elles s'associent aux pectines de la paroi, ils interviennent dans l'élongation cellulaire via MAPK3 chez *A. thaliana* (cf 2.1.4.1 L'élongation cellulaire) ; quand les WAKs sont en présence de fragments de pectines ou oligogalacturonides (OGs), générés par une lyse ou une destruction de la paroi, on observe une activation des MAPK3 et MAPK6 qui ont un rôle dans la défense face aux stress biotiques et abiotiques (Kohorn et al., 2012a).

Par ailleurs, grâce à l'utilisation d'une protéine chimérique constituée de la région extracellulaire d'AtWAK1 et du domaine kinase d'une autre protéine transmembranaire (EFR), il a été montré que l'apport exogène d'OGs induit l'activation du domaine kinase. Ceci confirme une réponse des WAKs aux OGs (Brutus et al., 2010). Cette liaison est facilitée par des conditions ioniques particulières (ratio calcium/cation monovalent) permettant la liaison des sucres dits en « boîte à oeufs » (Decrieux et Messiaen, 2005)

2.1.3.2 Interaction avec des protéines

La région extracellulaire de certains membres d'Arabidopsis, de riz et de peuplier présente un domaine EGF-like. Chez les animaux, ce domaine est présent dans les parties extracellulaires des protéines et jouerait un rôle dans l'interaction protéine/protéine (Stenflo et al., 2000). Une homologie entre les EGFs des animaux et les EGF-like des plantes amène à supposer un rôle similaire chez les plantes. Cette interaction protéine/protéine semble être modulée par le calcium (Stenflo et al., 2000).

D'autres études sont menées afin de trouver les acteurs protéiques impliqués dans la liaison avec les WAKs. L'une d'elle a permis d'identifier AtGRP3 (Glycine-Rich Protein), une protéine riche en glycine qui a la capacité d'interagir *in vitro* avec le domaine extracellulaire d'AtWAK1, AtWAK3 et AtWAK5 (Figure 2). Réciproquement, aucune des AtGRP2, 4, 6, 7 et 8 ne se lient au 5 isoformes WAKs (Park et al., 2001). La liaison entre GRP3 (ainsi que de son homologue structural GRP-3S) et WAK1 s'intègre dans une voie de signalisation de défense (cf 2.1.4.2 La réponse aux pathogènes).

2.1.4 Leurs rôles physiologiques

Dans la littérature, les WAKs semblent impliquées dans la réponse aux pathogènes ainsi que dans l'élongation cellulaire, chez Arabidopsis et le riz. On trouve également que certaines WAKs ont un rôle dans la réponse aux stress minéraux.

2.1.4.1 L'élongation cellulaire

La réduction de l'expression de toutes les WAKs chez un mutant d'Arabidopsis, par le biais d'un ARN antisens inductible, a mené à une perte d'expansion cellulaire (Kohorn 2001). En effet, tous les tissus où se trouvaient normalement les WAKs, ont montré une élongation cellulaire affectée (perte ou arrêt) et non pas d'effet sur la division cellulaire (Wagner et Kohorn, 2001, Lally et al., 2001).

On peut aussi observer, lorsque *AtWAK2* est muté (*wak2-1*), une perte d'expansion cellulaire au niveau racinaire dans des conditions limitées en sels et en sucres (Kohorn et al., 2006). Ces mutants *wak2-1* montrent une activité réduite de leurs invertases vacuolaires et une diminution de la turgescence cellulaire connue pour initier l'élongation cellulaire (Humphrey et al., 2007). En effet l'étude de ce mutant a mis en évidence la nécessité d'une liaison pectines/AtWAK2 pour activer l'accumulation d'invertase vacuolaire via AtMAPK3 (Kohorn et al., 2009). AtMAPK3 est une kinase cytosolique déjà connue par son implication dans les voies de régulation développementale et de réponses aux stress. L'ensemble de ces résultats suggère qu'AtWAK2 jouerait un rôle dans la régulation de la pression de turgescence et sur l'élongation cellulaire (Kohorn et al., 2006 ; 2009 ; 2012a)

Chez l'orge (*Hordeum vulgare*), *HvWAK1* est normalement exprimé dans les racines. Une mutation de celui-ci induit des racines plus courtes lors d'un stress salin (Kaur et al.,2013). Cependant les auteurs n'ont pas vérifié si ces racines plus courtes étaient la conséquence d'une réduction du nombre de cellules ou/et de la longueur de ces cellules.

2.1.4.2 La réponse aux pathogènes

Chez Arabidopsis, le gène *AtWAK1* est impliqué dans la mise en place des défenses vis-à-vis d'une attaque d'agents infectieux (He et al., 1998). En effet, une induction des ARNm WAK1 est observée lors de l'infection de pathogènes (ex : *P. Syringae*), ou après addition de composés comme l'acide salicylique (SA) ou d'un analogue structural 2,6-dichloroisonicotinic acid (INA). He et al. (1998) proposent un schéma de signalisation de la défense végétale (Figure 3). L'acide salicylique est perçu par la protéine NPR1(récépteur de SA) qui entraîne une régulation positive de protéines de défense (PR : Pathogen-Related) ainsi que celle de WAK1. De plus, la WAK1 peut moduler positivement sa

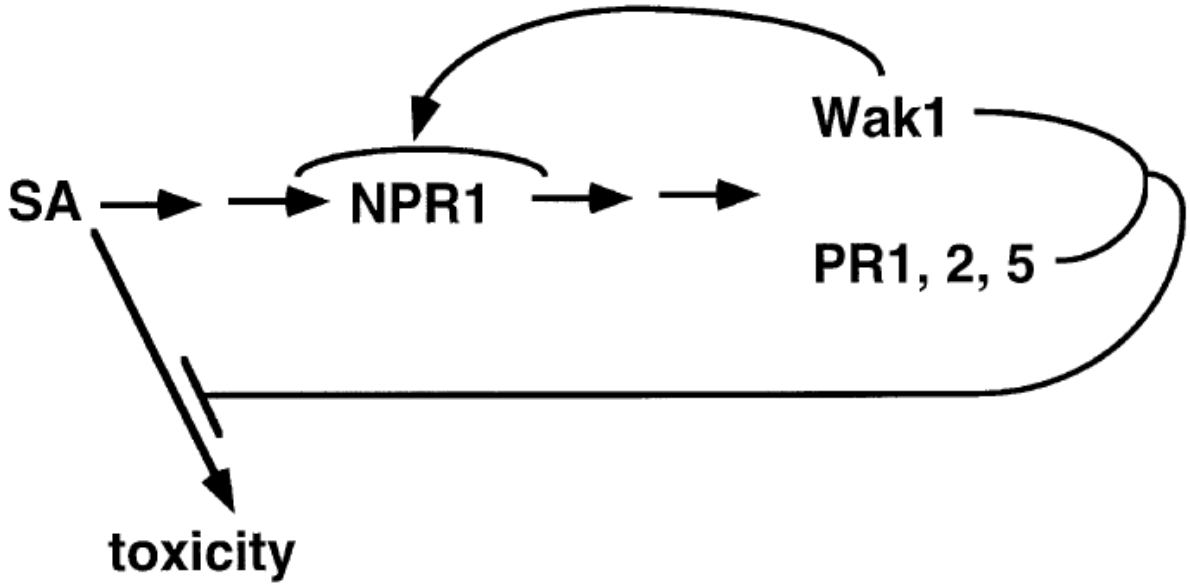

Figure 3. Schéma d'induction de WAK1 et son rôle dans la réponse à un pathogène d'après He et al., (1998). SA: acide salicilique, NPR1 : récepteur de SA , PR:protéine de pathogénèse

propre régulation en agissant en amont, en aval ou directement sur NPR1 (He et al., 1998). WAK1 et les protéines PR (PR1, 2 et 5) peuvent inhiber les effets toxiques de SA. De plus, SA induit la surexpression d'AtGRP-3 (Glycine-Rich secreted cell-wall Protein). La protéine WAK1 peut se lier à AtGRP-3 ainsi qu'à AtGRP-3S (protéine présentant 81% d'homologie avec AtGRP-3) avec son domaine extracellulaire (Park et al., 2001). L'ajout exogène d'AtGRP-3 dans le milieu contenant des protoplastes entraîne une augmentation de l'accumulation des transcrits et des protéines WAK1, PR1 et AtGRP-3 chez *A. thaliana* (Park et al., 2001).

Certaines WAKLs (AtWAKL1-7) peuvent intervenir dans la en réponse aux stress biotiques et abiotiques (Verica et al., 2003). Ce travail a montré une implication plus importante des AtWAKL5 et AtWAKL7 par rapport aux autres AtWAKLs, lors d'apport de SA ou d'INA et lors d'une blessure, suggérant leur implication dans les mécanismes de défenses, à l'instar de AtWAK1 (He et al., 1998).

2.1.4.3 La réponse minérale

Certaines données soutiennent l'idée que les WAKs interviendraient en réponse à un signal minéral bien que leurs modes d'action restent à découvrir.

L'utilisation d'un mutant *wakl4-1* d'*A. thaliana* a montré une hypersensibilité des plantes aux ions sodium (Na^+), potassium (K^+), cuivre (Cu^{2+}), nickel (Ni^{2+}) et zinc (Zn^{2+}), suggérant un rôle dans la signalisation minérale (Hou et al., 2005). Le mutant *wakl4-1* montre aussi une diminution de l'accumulation de Zn^{2+} dans la tige, indiquant que AtWAKL4 régule les gènes de transporteur de zinc (Hou et al., 2005).

Une autre étude affirme que AtWAK1 intervient dans une réponse rapide aux stress induits par l'Aluminium (Al) (Sivaguru et al., 2013). Un apport exogène d'Al induit une accumulation des ARNm d'AtWAK1 dans les racines. Dans les mutants sur-exprimant AtWAK1, une meilleure tolérance à l'Al est observée (Sivaguru et al., 2013).

2.2 Les séquences régulatrices

L'expression différentielle d'un gène, par exemple sous différentes contraintes, est déterminée en partie par les facteurs de transcription (FTs) et des éléments régulateurs. En effet la fixation et l'activation de l'ARN polymérase sont contrôlées par des FTs qui se fixent au niveau de séquences régulatrices (motifs) spécifiques. La région régulatrice primordiale qui assure le démarrage de la transcription est le promoteur dont la structure type sera développée dans une première partie. Le 1er intron, est également considéré comme une région d'ADN contenant des séquences régulatrices

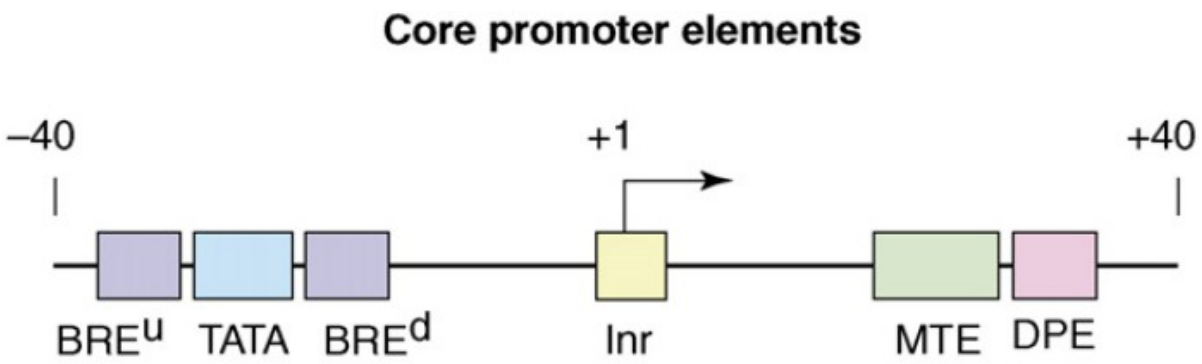

Figure 4: Schéma de la structuration d'un promoteur minimal. L'ensemble des motifs contribue à l'interaction entre l'ARN polymérase II et l'ADN. La TATA box, l'Inr, le MTE et le DPE participent à la reconnaissance de TFIID. Le BRE^u et BRE^d interagissent avec TFIIB d'après Tamar et al. (2008). BRE : B-Recognition Elements (u : upstream, d : downstream), DPE : Downstream Promoteur Element, Inr : Initiator, MTE : Motif Ten Element.

influençant la transcription et sera traité dans une seconde partie.

2.2.1 Le promoteur

Le promoteur eucaryote est une région de l'ADN génomique, non transcrite dans son intégralité, qui se situe en amont d'un gène et qui module l'expression de celui-ci. Son pouvoir modulateur dépend de deux critères ; l'état de condensation de l'ADN (Garcia et al., 2014) et la présence d'éléments régulateurs spécifiques. Un promoteur se structure en plusieurs parties : un promoteur minimal (ou core) de 80 pb entourant (-40 pb en amont et +40 pb en aval) le site de départ de la transcription (Transcription start site ou TSS) (Barrett et al., 2010 ; Danino et al., 2015), permet le recrutement et l'accrochage des différentes sous-unités de la RNA Pol II (TFII pour Trancriptor Factor of RNA PolII). En amont, on trouve le promoteur proximal (promoteur proche du promoteur minimal) et le promoteur distal (promoteur distant du promoteur cœur) (cf 2.2.1.2 Les éléments régulateurs) qui contiennent des éléments régulateurs.

2.2.1.1 Le promoteur minimal

Le promoteur minimal est défini comme la séquence minimale suffisante pour le bon fonctionnement de la machinerie transcriptionnelle (Butler et Kadonaga, 2002). Il comprend un ensemble d'éléments régulateurs essentiels à l'initiation de la transcription comme la TATA-box située autour de 30pb du TSS chez Arabidopsis (Molina et al., 2005). Ensuite il comprend également l'Initiateur (Inr), présent seul ou avec une TATA-box. Ces deux éléments servent de plateforme à l'assemblage du complexe de pré-initiation (PIC) de la transcription, par l'intermédiraire du TFIID (Roy et al., 2015). On trouve aussi l'élément de reconnaissance TFIIB (B-Recognition Elements ou BRE) placé en amont ($BRE^{upstream}$) et en aval ($BRE^{downstream}$) de la TATA-box, qui faciliterait la fixation de TFIIB au niveau du promoteur. D'autres éléments tels que le motif de dix éléments (Motif Ten Element ou MTE) et l'élément en aval du promoteur (Downstream Promoteur Element ou DPE) auraient une fonction de coopération avec l'Inr, et MTE peut même compenser l'activité basale de transcription en cas de perte de fonction du DPE ou de la TATA-box (Tamar et al., 2008) (Figure 4). La présence de tous les éléments n'est pas systématique dans le promoteur minimal, mais on observe plutôt une combinaison de ces différentes boites. Il existe aussi des promoteurs ne possédant pas de TATA-box.

De plus, deux types de fonctionnement du promoteur minimal existent naturellement et sont à l'origine des appellations promoteur minimal concentré et promoteur minimal diffus. Chez le promoteur minimal concentré l'initiation de transcription se fait soit en un seul site d'initiation au niveau du TSS soit au niveau d'un groupe distinct de TSS dans une courte région de quelques

nucléotides (Tamar et al., 2008). C'est le type de fonctionnement que l'on retrouve dans la plupart des organismes. Dans le promoteur minimal diffus, au contraire, il apparaît un certain nombre de sites d'initiation de la transcription observés sur une large région. Ceci est particulièrement notable chez les vertébrés avec la présences d'îlot CpG (Tamar et al., 2008).

2.2.1.2 Les éléments régulateurs

L'information génétique est identique et partagée par toutes les cellules d'un organisme, et pourtant l'expression d'un gène diffère selon le tissu où il s'exprime. Ceci s'explique en partie par la composition d'éléments régulateurs spécifiques du promoteur sur lesquels agissent les facteurs de transcription. La présence de ces séquences permettrait d'adapter la réponse d'un gène en fonction des besoins physiologiques. Il existe deux types de séquences régulatrices. Elles sont classées selon leur distance au site d'initiation de la transcription : les séquences proximales et distales.

- Les séquences régulatrices proximales

Les séquences régulatrices proximales peuvent avoir un effet activateur (UAS, pour upstream activating sequence) ou répresseur (URS, pour upstream repressing sequence) en fonction de l'activité de la protéine qui les reconnaît. La séquence la mieux étudiée est la boîte CAAT.

Dans ces régions proximales, les motifs situés entre -300 à -50 par rapport au départ de la transcription auraient un effet positif sur l'activité du promoteur, alors que ceux situés entre -1000 et -500 auraient une action négative sur l'activité du promoteur dans 55 % des cas testés (Barrett et al., 2010).

- Les séquences régulatrices distales

Les éléments distaux du promoteur peuvent être situés jusqu'à quelques milliers de paires de bases en amont ou en aval du site d'initiation et ils ont un impact sur la transcription. Les changements de la structure tridimensionnelle de l'ADN et de la chromatine favoriseraient le rapprochement entre un promoteur distal et le site d'initiation (Garcia et al., 2014). Ces éléments peuvent donc activer la transcription. Ils sont nommés 'enhancers'. *A contrario* les éléments réprimant la transcription sont nommés 'silencers'.

2.2.1.3 Chez Populus

Les connaissances de motifs chez *Populus* restent encore très fragmentaires. Des études d'un gène de régulation durant la formation du bois (PtxtMYB021) chez *Populus tremula x tremuloides*, ont montré au niveau du promoteur la présence d'un motif CCACCAAC (nommé ACTYP) qui est

similaire à des éléments AC impliqués dans l'activation des facteurs de transcription MYB pendant la biosynthèse de la lignine (Winzell et al., 2010). De même, chez *Populus euphratica*, 1182 pb de promoteur ont été isolés en amont du démarrage de la transcription du gène PeWRKY1. D'après Shen et al. (2015), l'analyse de ce promoteur révèle la présence de HSE (Heat Shock Element) en quatre répétitions en tandem dans la région régulatrice, qui seraient nécessaires à l'accrochage du facteur de transcription WRKY1, améliorant la tolérance à la salinité .

2.2.2 Le premier intron

A notre connaissance, les introns sont présents chez tous les organismes eucaryotes étudiés. Plusieurs théories sont présentées quant à leur origine. La première appelée « introns late » indique que le premier intron serait apparu après la séparation procaryotes/eucaryotes, la seconde théorie appelée « introns early » explique que LUCA (Last Univeral Common Ancestor) possédait déjà des séquences introniques (Jeffares et al., 2006). Les introns sont des séquences d'ADN transcrites avec les gènes en ARNs qui sont éliminées avant la traduction. Il a été montré que la longueur des introns était inversement corrélée avec le niveau de transcription (Barrett et al., 2012). Il semblerait que les gènes impliqués dans des

régulations rapides comme la réponse aux stress, le développement, la prolifération cellulaire possèdent moins d'introns que les autres gènes (Jeffares et al., 2006). La présence de motifs a été montrée au niveau du 1er intron chez Arabidopsis et d'autres espèces, indiquant son implication dans les mécanismes de régulation (IME pour Intron Mediated Enhancement) d'expression de gènes et que sa taille plus importante est dûe à un enrichissement en motifs (Bradnam et al., 2008). D'autres auteurs ont pu observer que le 1er intron peut se localiser dans le 5'UTR (35% dans le génome humain) (Barrett et al., 2012). Les introns possèdent des éléments régulateurs de la transcription, sont une source de séquences d'ARN non codées, participent à l'épissage alternatif (Barrett et al., 2012).

2.2.3 Motifs régulateurs des WAKs

Chez Arabidopsis, excepté *AtWAKL3*, tous les *WAKLs* présentent au niveau de leurs promoteurs, une boîte AS-1 (TGACG) requise pour que l'expression de PR1 soit induite en présence de SA (Lebel et al., 1998). En outre, de multiples copies d'une W-box (TTGAC) sont présentes chez les *WAKLs*, excepté chez *AtWAKL4*. Cet élément ressemble à la boîte AS, citée précédemment, et montre aussi l'induction possible par de l'acide salicylique (Verica et al., 2003 ; Meier et al., 2010) (cf 2.1.4.2 La réponse aux pathogènes). En particulier, *AtWAKL10* présente de nombreuses copies de la W-box, suggérant un rôle de ce gène en réponse aux stress abiotique et biotique.

Auteurs	Motifs
Hu	NAGAAN
	GATTA
	ATTTTCTTCA
Yamasaki	TNCGTACAA
Liao	TGACACA
	GTCAT
	CATCG
	CATTTG
Parra	CGATT
Xiao	GCTCA
	GAGAAGAATA
	AACGAC
	AATCTAATCT
	AAACAGA
	CAAATGAA
	ATTCTCTAAC
Winzell	CCACCAAC

Table 2: Liste de motifs retrouvés dans la littérature ajoutés à la base de données PLACE (Hu et al., 2014 ; Yamasaki et al., 2004 ; Liao et al., 2013 ; Parra et al., 2011 ; Xiao et al., 2014 ; Winzell et al., 2010).

L'étude d'*OsWAK11*, chez le riz, a montré au niveau de son promoteur, une multitude de motifs reliés à la réponse à différents stress. Par exemple, deux motifs CuRE (Cu Responsive Element, GTAC) sont impliqués dans la réponse au cuivre et à l'oxygène, une W-box est impliquée à la défense lors d'une blessure, une MeJA (MethylJasmonate)-responsive element motif (TGACG) participerait à la réponse durant un stress hydrique ainsi que beaucoup d'autres (Hu et al., 2014).

Les différentes études menées sur la famille multigénique des *WAKs* chez *A. thaliana, O. sativa et P. trichocarpa* présentent certaines de leurs caractéristiques moléculaires ainsi que certains de leurs rôles physiologiques. Cependant à notre connaissances les motifs régulateurs des *WAKs* chez *P. trichocarpa* n'eut pas été présentés dans la bibliographie. Mon travail a porté sur le développement d'une stratégie d'étude *in silico* des motifs régulateurs des WAKs chez *P. trichocarpa* et sur la recherche de corrélations entre la présence de ces motifs et les profils d'expression des membres de cette famille multigénique. Les résultats obtenus autours de cette analyse exploratoire sont présentés dans ce rapport.

3.Matériels et Méthodes

3.1. Les bases de données

3.1.1 Le génome de *P. trichocarpa*

Le génome de *P. trichocarpa* a été séquencé pour la première fois en 2006. Aujourd'hui, c'est la 3eme version (V3) d'annotations qui est utilisée. La base de données du PIAF (PIAFdb) (Piafdb@univ-bpcermont.fr) possède cette génération d'annotation du génome de *P. trichocarpa* et a été utilisée pour les recherches de motifs.

3.1.2 Base de données des motifs régulateurs

PLACE (PLant Acting Cis Element) est une base de données de motifs, disponible sur internet (Higo et al., 1999) (http://www.dna.affrc.go.jp/htdocs/PLACE/). Cette base de données regroupe un total de 420 motifs identifiés et caractérisés chez les plantes. Pour ma recherche, j'ai complété cette base de données par d'autres motifs publiés (Hu et al., 2014 ; Yamasaki et al., 2004 ; Liao et al., 2013 ; Parra et al., 2011 ; Xiao et al., 2014 ; Winzell et al., 2010) (Table 1).

Figure 5 : Photo d'expérimentation gravitropique.. Dispositif permettant l'inclinaison des peupliers dans un environnement lumineux isotrope (ANR TROPIC)

3.2 Les langages de programmation

3.2.1 Le Perl

Le Perl est un langage informatique optimisé pour l'extraction d'information de fichier texte et pour le développement de script. Ce langage a été conçu pour manipuler les chaînes de caractères et pour la création d'outils. Il a été utilisé ici pour créer les outils nécessaires à la détection des motifs régulateurs le séquences promotrices et introniques.

3.2.2 Le Bash

Le Bash (l'acronyme de Bourne-Again SHell) est un interpréteur en ligne de commande. Il facilite le traitement des flux de données. Le langage bash est utilisé avec une version Linus/Ubuntu.

3.3 Le logiciel R studio

R studio est un système dit langage-logiciel, puisqu'il nécessite un langage de programmation pour le traitement des données. C'est un environnement de travail mathématique principalement utilisé pour l'analyse statistique. Le logiciel Rstudio est gratuit avec deux versions disponibles : une version permet une exécution locale du logiciel, l'autre version permet d'accéder à un serveur *via* un navigateur web. C'est de la deuxième version, Rstudio Server, qui est utilisée dans notre étude (R version 3.3.0 (2016-05-03)).

3.4 Jeu de données

3.4.1 qPCR « organes » :

Les échantillons d'organes ont été générés par K. Tocquard (thèse 2016) à partir de peupliers hybrides (*Populus tremula x alba*, clone INRA '717-1B4') placés en hydroponie. Différents organes comme la tige, le pétiole et le limbe en croissance, ainsi que l'apex, ont été prélevés à l'entrenœud 38 de peupliers présentant 43 entrenoeuds développés. Le pétiole et le limbe matures, les bourgeons ainsi que différents tissus tels que l'écorce et le xylème matures ont été prélevés à l'entrenœud 10. Les résultats d'expression ont été obtenus par PCR quantitative en temps réel (K. Tocquard, thèse 2016).

3.4.2 qPCR Gravitropique « Isotropique » :

Ces données sont issues du programme ANR TROPIC du ministère de la recherche. L'objectif du projet était d'étudier l'effet de la gravité sur les plantes dans un environnement lumineux isotrope. Dans ces expériences, les plantes ont été placées dans une sphère associée à un dispositif de lumière isotrope (Figure 5), et ont été inclinées à différents temps (30 min, 1h, 2h et 3h) à 15° et 30° degrés par rapport à la verticale). Le xylème est extrait, en séparant avec un scalpel la partie supérieure et

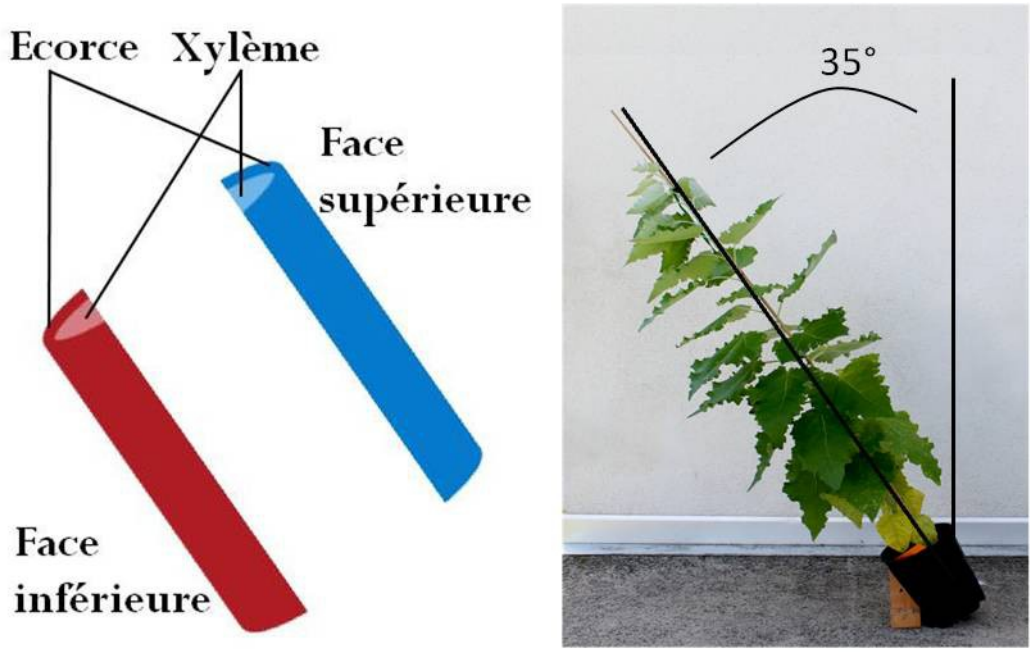

Figure 6 : Mode d'inclinaison des peupliers en serre et schéma de prélévement des tissus. Les peupliers ont été inclinés à 35° par rapport à l'axe vertical grâce à une cale placée sous le pot. Les entrenoeuds de la partie basale de la tige ont été prélevés, les faces supérieures et inférieures ont été séparées à l'aide d'un scalpel. Les tissus de l'écorce et du xylème ont été séparés de chaque ½ cylindre de tiges.

inférieure de la tige par rapport à l'axe d'inclinaison. Les résultats d'expression ont été obtenus par PCR quantitative en temps réel en utilisant des plaques micro-fluidigm 96x96 de la société Fluidigm (Fluidigm, USA) (Lopez, post doctorat 2014).

3.4.3 qPCR Gravitropique « Normale » :

Cette expérience a été réalisée par K. Tocquard lors de sa thèse (en cours d'édition, 2016) dans le cadre des recherches de l'équipe MECA. L'objectif de cette expérience était d'étudier la réponse des gènes *WAKs* à l'effet gravitropique. Les plantes utilisées (*Populus tremula x alba*) ont été inclinées à 35°degrés par rapport à l'axe vertical à l'aide d'un tuteur (Figure 6), à 3 temps différents (30, 60 et 120 min). Deux types de tissus ont été échantillonnés ; le xylème (faces supérieure et inférieure) et l'écorce (faces supérieure et inférieure). L'ensemble des échantillons a été étudié par PCR quantitative en temps réel.

3.4.4 qPCR de turgescence foliaire

Le jeu de données a été obtenu à l'issue d'expérimentations que j'ai réalisées en collaboration avec P. Benoît (Master 2 GEPV 2016).

-Matériel végétal et condition de stress

Des clones de peupliers (*Populus tremula x alba*) ont été placés en serre pour un contrôle optimal des conditions du milieu. Une partie des plantes a été placée en carence hydrique progressive jusqu'à un état sévère, puis ré-arrosées. Les feuilles ont été prélevées dans une gamme de potentiel hydrique (-5, -10, -20, -30 bars pour la sécheresse et -8, 0 bars pour le ré-arrosage). L'étude des transcrits est effectuée, après extraction et génération d'ADNc, ce travail a été réalisé par P. Benoît (Master 2), sur les deux parties de la feuille (limbe et nervure). J'ai ensuite réalisé la PCR quantitative en temps réel.

-PCR quantitative en temps réel

1,5µl par échantillon, dilués au vingtième, sont mélangés à une solution tampon (7,5µl) et aux amorces des gènes étudiés (température d'hybridation 56°C, N Cycle) (0,5 µl sens et anti-sens). L'amplification est réalisée par un thermocycler Bio-Rad iQ5. Leur expression a été évaluée par la méthode du 2ΔΔCt (Pfaffl, 2001) en la rapportant à l'expression de gènes de référence tels que ACT1 (actine), EFR1 et UBQ146 (ubiquitine).

3.4.5 RNAseq de Stress hydrique :

Six génotypes de peupliers noirs (*Populus nigra*) ont été placés en condition de carence hydrique contrôlée. De 8 à 10 jours après l'initiation du stress hydrique, 3 feuilles (de rang 5) par arbre ont été

MotifSequence	TargetID	MotifPosList	Counts
CTCTT	Potri.001G038300	(970)	1
CTAACCA	Potri.017G117400.1	(228)	1
TGAGTCA	Potri.007G125100	(66;89;521)	3
TGTCA	Potri.001G038300	(860)	1
GTGA	Potri.T132900	(244;387;792)	3
GAAAAA	Potri.010G120800.1	(39;55;117;182;343;425;481;802)	8
AACCAA	Potri.001G038300	(230;234;776;879)	4
TATAAAT	Potri.001G038300	(900)	1
CATGCAY	Potri.T167400.1	(936)	1

Table 4: Extrait de résultats obtenus avec l'outil 'detectMotifs'. "MotifSequence" montre la séquence de chaque motif, "TargedID" indique le nom du gène où il a été retrouvé. Si le motif est trouvé dans le 1er intron, le nom du gène comporte une extension (.1). Lorsque cette extension est absente, le motif est localisé dans le promoteur. "MotifPosList" localise les motifs sur les séquences identifiées. "Counts" indique le nombre de fois que le motif est compté dans la séquence.

prélevées à un état physiologique identique pour tous les arbres (-20 bars). Les échantillons de limbes (feuilles sans nervure centrale) ont été analysés, après extraction des ARNm, en RNAseq avec la méthode de paired-end (Illumina Hiseq 2000, Konstanz). Ce travail a été réalisé par M. Garavillon.(en thèse à l'UMR PIAF)

L'ensemble des calculs et des gestions des données a été effectué sur le serveur du PIAF (PIAFdb) (http://www.piafdb.univ-bpclermont.fr).

3.5 Étude *in silico* des promoteurs et 1er intron

3.5.1 Obtention des séquences promotrices et introniques

L'outil 'DownloadPIAFdbPromoteurs' (voir annexe N°1) permet d'obtenir les séquences promotrices de chaque modèle de gènes présents dans une liste de données. De la même façon pour les introns, l'outil 'getSequencesFromPIAFdb' (voir annexe N°2) récupère toutes les séquences introniques à partir d'une liste de gènes. Lors de la requête, le premier intron peut être extrait en le précisant lors de l'exécution de l'outil. Ces outils et une méthodologie identique ont été utilisés pour les recherches sur l'ensemble des gènes WAK d'une part, et pour le reste du génome de *P. trichocarpa* (hors WAK) d'autre part.

3.5.2 Recherche de motifs

La recherche de motifs est réalisée sur la base de motifs déjà connus et caractérisés dans la base de donnée PLACE ((http://www.dna.affrc.go.jp/htdocs/PLACE/) complétée (cf 3.1.2 Base de données des motifs régulateurs). L'outil 'detectMotifs' (voir annexe N°3) permet, en lui fournissant une liste de motifs la plus exhaustive possible, de détecter pour chaque motif, l'identifiant du gène où il a été retrouvé, la localisation sur les séquences soumises (promoteur et 1er intron) ainsi que le nombre de fois qu'il a été trouvé pour chaque modèle de gène au niveau du promoteur et du 1er intron. La détection de motifs est effectuée dans la famille des PtWAKs (Table 4) ainsi que dans le reste du génome de *P. trichocarpa*.

3.5.3 Analyse des motifs et tests statistiques

L'analyse des motifs est réalisée avec l'outil 'analyseMotifs' (voir annexe N°4). Cette analyse s'effectue avec les données obtenues par l'outil 'detectMotifs' (l'une avec la famille des WAKs et l'autre avec tout le reste du génome). Le rôle de cet outil est d'analyser, plus précisément, les motifs

motifSeq	motifLength	targetIDs	posLists	countsLists	nbTargets	allTargets	targetFreq	meanMatch	varMatch	medianMatch	minMatch	maxMatch	nbMatches	allMatches	matchProp
NAGAAN	6	001G038300.2;Potri.001G038300.3;Potri.	338;572;635)_(74;7	;12;11;10;5;9;	371	383	0.969	6.83	37.87	6.00	1	52	2533	40544	0.062
AAAG	4	tri.T136600;Potri.T136700;Potri.T136900;P	70;366;973)_(222;47	;9;8;11;5;7;10	358	383	0.935	6.72	27.57	6.00	1	44	2404	40544	0.059
YACT	4	T136700;Potri.T136800;Potri.T136900;Pot	946;979)_(659)_(18	6;6;7;15;9;6;4	357	383	0.932	5.90	17.74	5.00	1	24	2108	40544	0.052
CAAT	4	32900;Potri.T136400;Potri.T136500;Potri.T	3;616;623;785)_(20	;11;7;3;2;3;9;6	355	383	0.927	5.99	25.75	5.00	1	41	2128	40544	0.052
NGATT	5	0;Potri.T132800;Potri.T132900;Potri.T136	489)_(450;596;652;6	2;6;7;2;3;4;2;6	352	383	0.919	5.23	16.84	5.00	1	30	1840	40544	0.045
GRWAAW	6	2800;Potri.T132900;Potri.T136400;Potri.T1	38;439)_(377;508;6	2;5;5;9;8;9;3;10	350	383	0.914	4.65	16.31	4.00	1	34	1628	40544	0.040

Table 5: Extrait de résultats obtenus avec l'outil 'analyseMotifs'. Parmi l'ensemble des résultats, on retrouve la séquence du motif, sa longueur, l'identifiant des cibles où il a été localisé, la position sur les différentes séquences analysées, le nombre total de séquences étudiées.

motifSeq	motifLength	testTargetFreq	refTargetFreq	fdrAdjPvalue	trend
GGATA	5	0.376	0.020	0	over
TGACT	5	0.504	0.080	7E-048	over
AWTTCAAA	8	0.225	0.020	7E-045	over
TTGACC	6	0.217	0.020	2E-041	over
CWWWWWWWWG	10	0.467	0.080	4E-040	over
GTCAT	5	0.441	0.080	7E-035	over
TGACY	5	0.653	0.160	1E-032	over

Table 6: Extrait de résultats obtenus avec l'outil 'testMotifs'. On retrouve la séquence du motif, sa longueur, la P-value ajustée de la correction de Benjamini-Yekutieli, et la sur-(over) ou sous-(less) représentation dans la famille de PtWAKs

trouvés par l'outil 'detectMotifs'. Parmi les résultats obtenus par 'analyseMotifs', avec l'option « allDetails » on retrouve : la séquence des motifs, leurs longueurs, l'identité de tous les gènes (promoteur et intron confondus) présentant au moins une fois ce motif, la liste de leurs positions dans les séquences, le comptage des motifs pour chaque gène, le nombre de séquences où le motif a été retrouvé, le nombre total de séquences analysées (Table 5). Une fois cette analyse réalisée chez les WAKs et dans le reste du génome, l'outil 'testMotifs' (voir annexe N°5) permet de ressortir les motifs sur-représentés chez les WAKs. L'outil 'testMotifs' ne prend en compte que le format en sortie de l'outil 'analyseMotifs' avec l'option « allDetails ». L'outil 'testMotifs' effectue un test de X^2 entre l'abondance des motifs dans la famille étudiée et l'abondance des motifs dans la référence (le génome du peuplier sans la famille des WAKs), avec une P-value ajustée d'après Benjamini-Yekutieli (2001). Les motifs qui en ressortent sont sur-représentés dans cette famille par rapport au reste du génome de *P. trichocarpa* (Table 6).

3.5.4 Matrice de fréquences

Les données obtenues précédemment permettent de générer une matrice de fréquence. Pour un gène donné, la fréquence calculée pour chaque motif spécifique (Fm) est la somme du nombre de fois que ce motif est retrouvé dans le promoteur (Pm) et le nombre de fois compté dans le 1er intron (Im) sur l'ensemble des séquences analysées (383) (Fm=(Pm+Im)/383).

3.6 Étude de corrélation entre motifs et jeu de données

L'étude des corrélations est réalisée avec Rstudio. Les données sont issues de la matrice de fréquences (voir précédemment) auxquelles s'ajoutent les résultats des différents jeux de données décrits précédemment (cf 3.4 Jeu de données). Cette étude est effectuée en Analyse en Composante Principale (ACP) pour chaque jeu de données. Les résultats sont obtenus graphiquement, présentant simultanément les individus (gènes) et les variables (motifs et modalité de stress). On fixe une limite pour la qualité de représentation (cos^2=0,3) et on prend une valeur de contribution théorique relative (contrib=1/√n, avec n=nombre d'individu ou de variable) des gènes et des variables.

4.Résultats

4.1 L'enrichissement des motifs chez la famille des *PtWAKs*

Sur l'ensemble de la recherche de motifs effectuée dans les séquences promotrices et introniques chez la famille multigénique de *Pt*WAKs, 112 motifs sont retrouvés sur-représentés spécifiquement

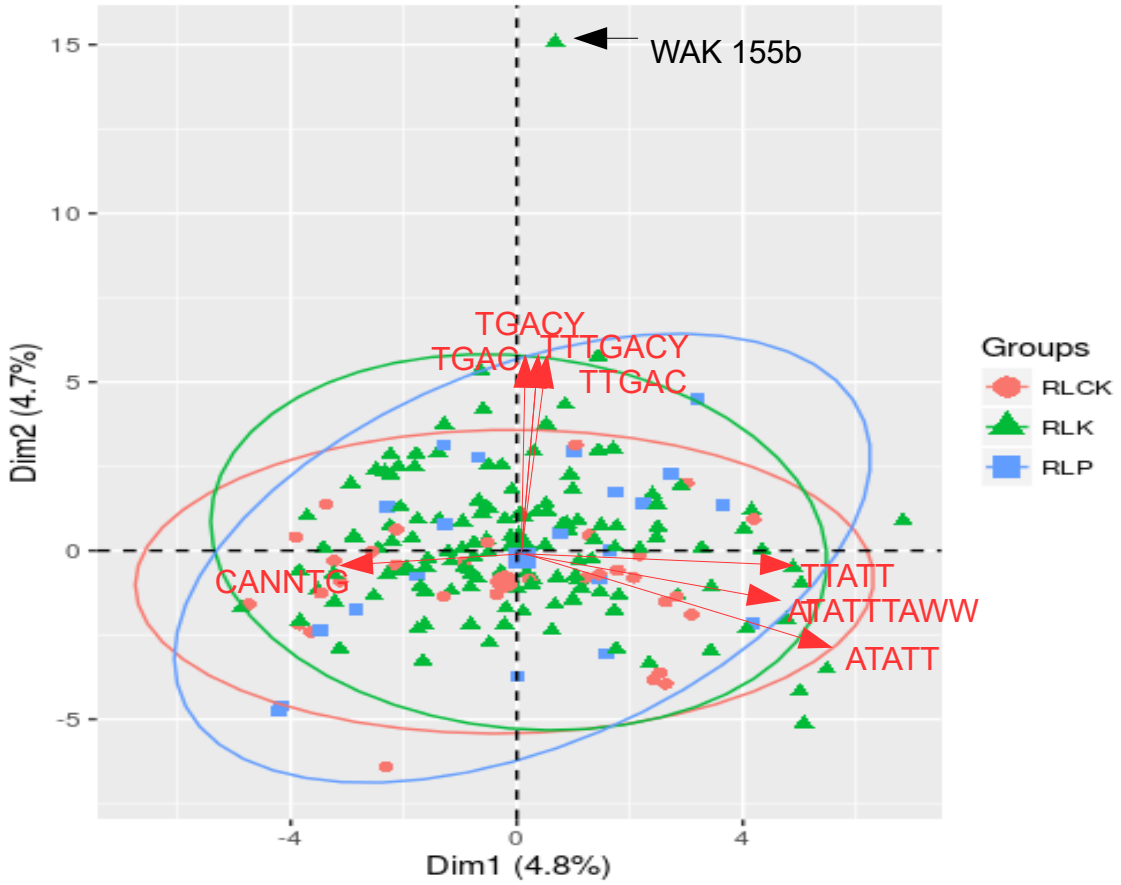

Figure 7: Analyse en composante principale (ACP) de la fréquence des 112 motifs par les 175 gènes de PtWAKs. A : Adénine, T : Thymine, C : Cytosine, G : Guanine Y : C ou T, W : A ou T, RLK : Recepteur-Like Kinase, RLCK : Recepteur-Like Cytoplasmic Kinase, RLP: Recepteur-Like Protein.

chez cette famille après le test de Khi² effectué par l'outil 'testMotifs'. Les motifs sélectionnés ont une P-Value inférieure à 5%, ce qui confirme que leurs sur-représentations chez les *Pt*WAKs soient probablement peu liées au hasard. Certains motifs présentent des séquences plus ou moins dégénérées (*i.e* TGACY ou CWWWWWWWWG).

4.2 Analyse des relations entre les motifs et les *PtWAKs*

L'analyse de l'ACP est effectuée à partir du tableau de fréquences décrit précédemment (cf 3.5.4 Matrice de fréquences). La valeur propre correspond à la proportion de la variance totale représentée sur l'axe. La décroissance des valeurs propres a été analysée pour choisir les 2 dimensions représentant le maximum d'information sur les 2 axes. Ainsi le plan montré est celui qui donne le maximum d'information représenté par les deux axes (Figure 7). Dans une ACP, le nombre de dimension est égal au nombre de variables utilisées lors de l'analyse. Ceci explique, en partie, le faible pourcentage d'information présenté par les différentes ACP (Figure 7) réalisées à partir d'un nombre de motifs importants (112). Les résultats présentés prennent en compte les dimensions qui maximisent la variance. Afin de donner un poids aux suppositions qui sont émises, on ne représente que les individus et les variables avec une qualité de représentation (cos²) supérieure au seuil fixé (cf 3.6 Étude de corrélation entre motifs et jeu de données). De même, on ne représente que les variables et individus avec une contribution supérieure à la contribution théorique relative (cf 3.6 Étude de corrélation entre motifs et jeu de données) et ceci pour l'ensemble des données générées par l'ACP. Le premier axe donne seulement 4,8% de l'information et le second axe 4,7% ; l'information totale représentée par l'ACP est de 9,5% (Figure 7). En ne prenant en compte que la qualité de représentation (cos²), 2 paquets de motifs sont corrélés et participent à la formation des 2 axes. Les motifs TTATTT, ATATT et ATATTTAWW sont bien représentés par le premier axe. Les motifs composés de TGAC et de ses variants (TGACY, TTTGACY et TTGAC) sont bien représentés sur le deuxième axe. Il en est de même pour le motif CANNTG qui est bien représenté sur l'axe 1. Pour soutenir ces propos, la comparaison avec l'ACP des contributions des variables (Annexe) est nécessaire. Les résultats obtenus avec les contributions corroborent la formation des axes par les motifs représentés. Les variables qui représentent les axes sont orthogonales entre elles. Les motifs TTATTT, ATATT, ATATTTAWW et TGAC, TGACY, TTTGACY, TTGAC ne sont pas corrélés. Le motif CANNTG est anti corrélé aux motifs TTATTT, ATATT, ATATTTAWW (ils sont sur le même axe, mais les vecteurs sont de sens opposés). Autrement dit, le motif CANNTG n'est pas présent sur les gènes contenant les motifs TTATTT, ATATT, ATATTTAWW et inversement.
On remarque aussi que la *WAK155b* est bien représentée par le deuxième axe. On suppose d'après l'ACP que le motif TGAC et ses variants participent, en partie à la formation des séquences

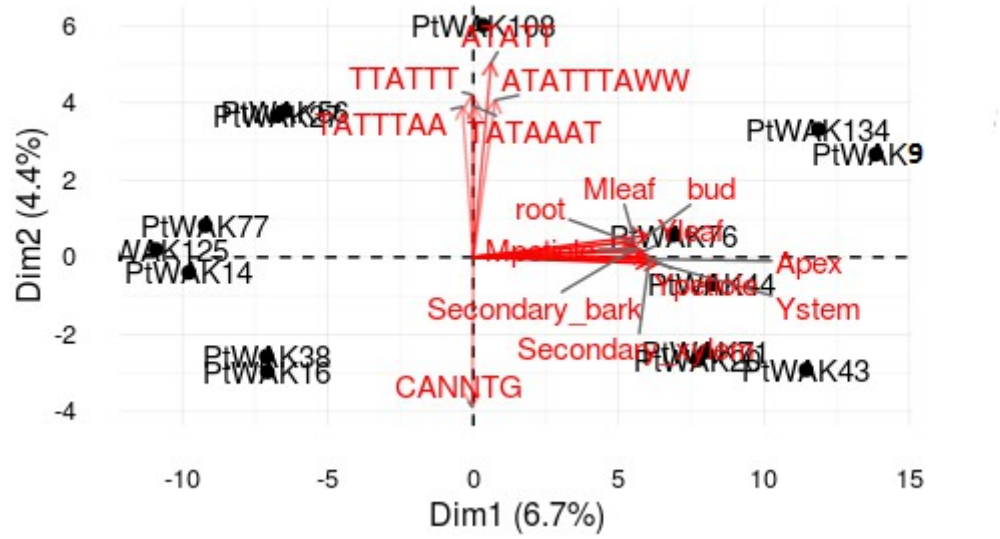

Figure 8: ACP des fréquences de motifs et d'expression relatives des WAKs dans les organes de P. trichocarpa. Bud : bourgeon, Roots : racines, Mleaf : feuille mature, Yleaf : jeune feuille, Y stem : jeune tige, Secondary_bark : écorce secondaire, Secondary_xylem : xylème secondaire.

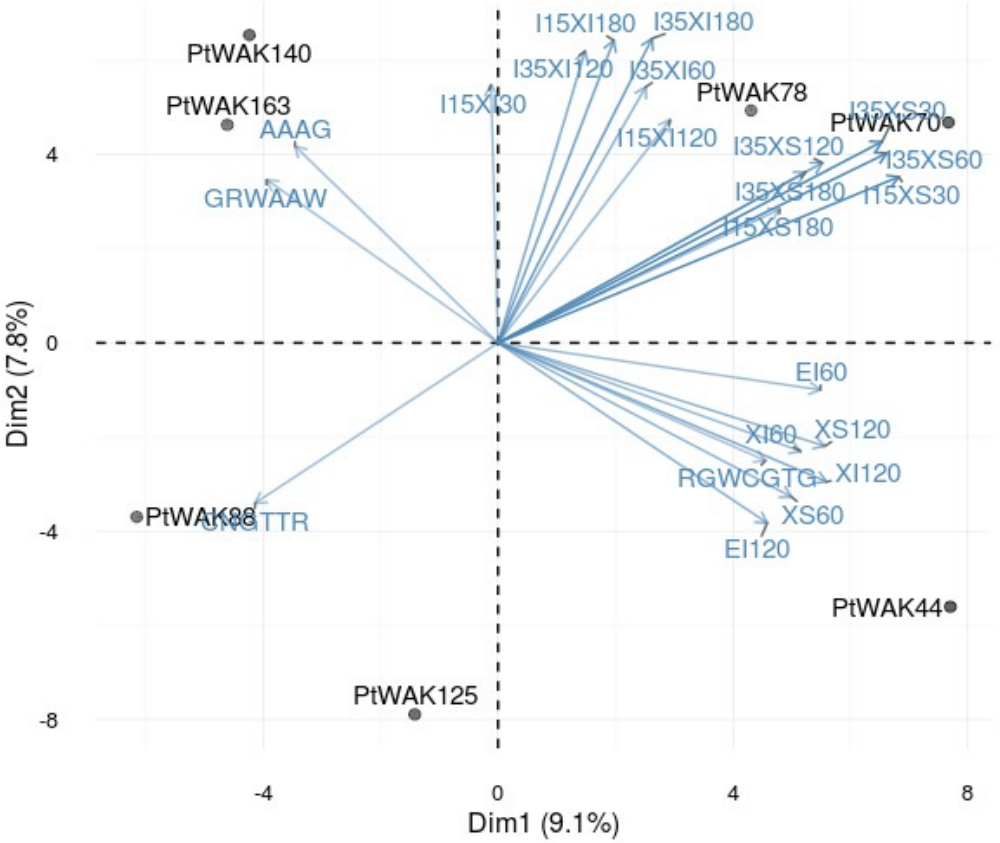

Figure 9: ACP des fréquences de motifs associées aux stress gravitropiques. I : Isotropique, XI : xylème inférieure, XS : xylème supérieure, EI : écorce inférieure, 15 et 35 : degrés d'inclinaison, 30, 63, 120 et 180 : temps d'inclinaison en minute.

régulatrices chez la *WAK155b*.
Dans l'ACP, les *WAKs* sont regroupées par type de gènes (RLK, RLCK et RLP). Les ellipses de confiance sont tracées pour chaque type de gènes, avec au centre le point d'inertie de la dispersion des gènes. Ces ellipses indiquent avec 5% d'erreur que l'ensemble des types de gènes se situe dans ces ellipses. Le chevauchement des 3 ellipses indique que les types de *WAKs* ne diffèrent pas en fonction de leurs fréquences en motifs et aucun motif ne structure spécifiquement un type de *WAKs*.

Le tableau de fréquence de motifs est complété par le jeu de données d'expression des *WAKs* dans les organes de *P. trichocarpa* (cf 3.4.1 qPCR « organes »). L'ACP (Figure 8) indique que la formation de l'axe 1 est due aux différents organes et que les motifs représentés participent à la formation de l'axe 2. On retrouve la même anti-corrélation entre les motifs TTATTT et ses variants et le motif CANNTG que précédemment. Les motifs représentés sur le deuxième axe ne sont pas corrélés avec l'expression des *WAKs* dans les différents organes. On observe qu'un cluster de *WAKs* composés des *WAKs 134, 90, 76, 44, 71, 25* et *43* est bien représenté dans les organes. Mais en regardant l'ACP des contributions (annexe n°6), on remarque que le cluster cité précédemment n'est pas tout à fait confirmé. On observe en effet que les *PtWAKs 134, 90, 71 et 43* ont plus de poids sur le second axe. En comparant les deux, on peut supposer que l'expression de *PtWAKs* 134, 90, 71 et 43 pourraient hypothétiquement être corrélée aux organes étudiés.

4.3 Analyse des corrélations entre les motifs et le « stress gravitropique »

Les données des 2 expériences de stimulation gravitropique sont traités simultanément. seules les *PtWAKs* dont l'expression est modulée sont prises en compte pour l'ACP, soit 32 *PtWAKs* au total.
Parmi ces 32 on dénombre 76 séquences régulatrices. Ceci signifie que la fréquence des motifs (Fm) n'est pas identique à celle utilisée précédemment: soit Fm=Pm+Im/76.
L'ACP (Figure 9) montre 17% de l'information totale. Les variables des stress gravitropiques sont regroupées en 3 lots. Le premier lot comporte des variables des données gravitropiques en condition isotrope, qui sont bien représentées sur l'axe 2. Toutes ces variables indiquent que les échantillons XI (Xylème inférieur) sont corrélés entre eux, à l'exception de I15XI30. Le second lot indique que les échantillons XS (Xylème supérieur) sont tous corrélés entre eux. Les variables du troisième lot, issues des données gravitropiques en condition normale semblent aussi être corrélées entre elles.
On peut observer que le motif RGWCGTG est colinéaire aux variables EI60, EI120, XI120, XI160, XS120 et XS160. Ceci suggère une corrélation entre le motif et le stress gravitropique en condition normale. Le gène *PtWAK44* est colinéaire aux stress gravitropiques en condition normale. Ceci

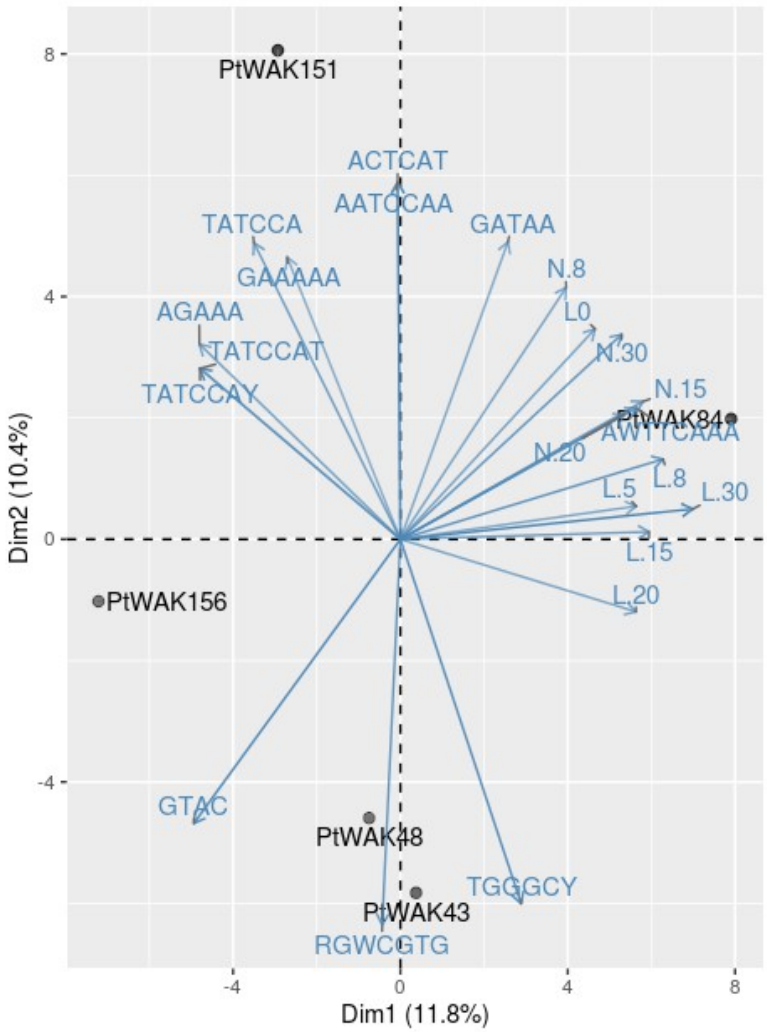

Figure 10: ACP de 19 PtWAKs associées à l'état de turgescence foliaire. L : limbe, N : Nervure, potentiel hydrique:-5, -10, -15, -20, -30, -8, 0 (en bars).

suggère que l'expression de *PtWAK44* est corrélée aux stress gravitropiques en condition normale. Le groupe composé de *PtWAK163* et *140* et des motifs AAAG et GRWAAW sont anti-corrélés aux stress gravitropiques en condition normale et à *PtWAK44*.

Les *PtWAK78* et *70* sont regroupées au niveau des variables de stress gravitropiques en condition isotrope et plus particulièrement *PtWAK70.* L'expression des *AtWAK78* et *70* semble être liée à aux stress gravitropiques dans un environnement isotrope. On observe que *PtWAK88* est corrélée avec le motif CNGTTR, et *PtWAK163* et *140* avec les motifs AAAG et GRWAAW. *PtWAK88* et CNGTTR sont anti-corrélés aux stress gravitropiques en condition isotrope et aux *PtWAK70* et *78*.

4.4 Analyse des corrélations entre les motifs et « la turgescence foliaire »

Dix neufs *PtWAKs* ont été suivis durant cette expérience. Ici Fm= Pm+Im/44, 44 étant le nombre des séquences analysées de l'échantillonnage. L'information totale transmise par l'ACP est de 22% (Figure 10) Le premier axe (11,8% d'information) est en grande partie structuré par les différentes échantillons prélevés à des potentiels de pression choisis et par tissu (limbe, nervure). On observe aussi que le motif AWTTCAAA participe à la formation du premier axe et est corrélé aux différents potentiels de pressions. Ceci suggère que le motif AWTTCAAA aurait un relation probable avec l'état hydrique de la feuille. Les autres motifs présents sont soit non corrélés au stress comme les motifs ACTACT, AATCCAA ou RGWCGTG qui participent à la structuration de l'axe 2 (10,4% d'information), soit anti corrélée comme le motif GTAC. Le gène, *PtWAK84* est colinéaire, sur l'axe 1, au stress et au motif AWTTCAAA, ce qui suggère son implication dans ce stress et avec une structuration régulatrice liée au motif AWTTCAAA. Le gène *PtWAK156* participe à la structuration de l'axe 1, et est anti corrélé au stress hydrique.

4.5 Analyse des corrélations entre les motifs et le « stress hydrique »

La représentation des individus dans les plans principaux est seulement de 9,7% (Figure 11). Les gènes sont représentés en fonction de leur modulation lors du stress hydrique. Le gène est soit « UP » pour sur-exprimé, « ND » pour non différentiel et « Down » pour les sous-exprimé. La dispersion des gènes sur le graphe ne permet pas de l'interpréter correctement. Par conséquent, le travail d'analyse est effectué à partir des barycentres de chaque groupe de gènes. Une ellipse de confiance est ajoutée, avec un taux d'erreur de 5%. Les ellipses des groupes « ND » et « DOWN », se chevauchent, indiquant qu'aucune fréquence de motifs ne permet de différencier ces deux groupes. On observe que le groupe « UP », chevauche légèrement l'ellipse « ND ». On remarque aussi une tendance se dessiner pour les gènes « UP » à se retrouver sur le premier axe. Le barycentre « ND » et « DOWN » se retrouvent confondus au point d'origine des axes. Les gènes

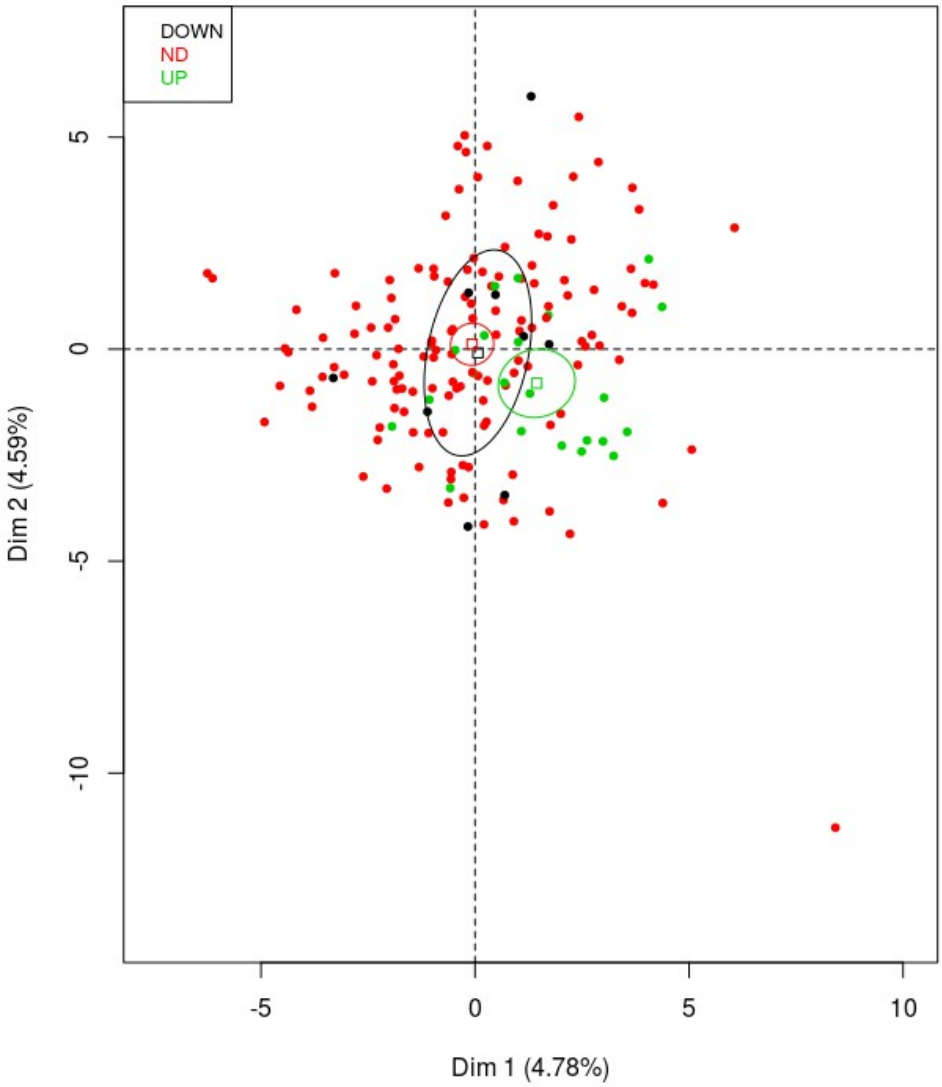

Figure 11: ACP des PtWAKs en fonction de leurs modulations d'expression par le stress hydrique. « Down » : sous expression par rapport à un témoin hydraté. « ND » : pas de modification d'expression. « UP » : surexpression par rapport à un témoin hydraté.

« UP » sembleraient avoir une tendance à s'éloigner de l'origine dans la direction du premier axe.

5. Discussion

5.1 Les motifs chez *PtWAKs*

L'utilisation d'outils bio-informatiques et statistiques a permis l'identification de cent douze motifs repartis le long des régions régulatrices du promoteur et de premier intron. Nous n'avons pas pu mettre en évidence l'existence de corrélation entre la fréquence de motifs et l'appartenance des gènes à une famille kinase particulière RLK, RLCK ou RLP. Cependant, *PtWAK155*b a montré une structuration particulière au niveau de son promoteur par la forte représentation du motif TGAC. Ce motif se nomme W-box. Il participerait à la mise en place des réponses de défense cellulaire lors d'une blessure (Meier et al., 2012). Une diminution des défenses cellulaires chez le mutant permettrait de confirmer le rôle du gène *PtWAK155b* dans ces réponses. Des expériences de mutagenèse dirigé pour l'élimination de ce motifs TGAC dans le promoteur et l'analyse des mutants obtenus soumis à des blessures, permettrait de confirmer la fonction des W-box pour le gène *PtWAK155b*. D'après Tocquard et al. (2014) 42 *PtWAKs* sont modulées en réponse à différentes conditions de stress, biotique (pathogène) et abiotique (sécheresse). Parmi les *PtWAKs* étudiées, 7 semblent répondre à une attaque de pathogènes, *PtWAK155b* ne faisant pas partie de cette liste. Une étude comparative des séquences promotrices des 8 *PtWAKs* (incluant *PtWAK155b*) pourrait permettre de trouver un motif partagé entre les *PtWAKs* modulées par des blessures causés par un pathogène. Cependant de manière troublante, les *PtWAKs* régulées par une attaque de pathogènes (Tocquard et al., 2014) ne sont pas représentées au sein de l'ACP (Figure 7). En tenant compte que l'information transmise par l'ACP n'est que de 9,5%, il est essentiel de confirmer ces résultats obtenus par des méthodes expérimentales complémentaires.

La recherche de corrélations entre les fréquence des motifs régulateurs et les données d'expression des *PtWAKs* dans les différents organes analysés n'a rien révélée. Ce résultat n'est peut être pas très surprenant car les motifs régulateurs utilisés ne sont pas caractérisés comme des motifs modulant l'expression spécifiquement dans un organe. Il est possible qu'une recherche de motifs *de novo* permettrait de découvrir un motif permettant une expression « tissus/spécifique ».

5.2 La relation entre les motifs et le stress gravitropique

L'étude de l'analyse en composante principale de l'expression relative des *PtWAKs* à une stimulation gravitropique comportait de 2 jeux de données issus de 2 expérimentations différentes. L'une dans un environnement isotrope et l'autre en condition normale. L'ACP met en évidence une séparation des deux jeux de données. Les échantillons isotropes sont corrélés positivement aux 2 plans, alors que la stimulation gravitropique en condition normale est corrélée positivement au premier axe et négativement au second. De manière attendue, les contributions des différentes variables (annexe n°6), sont aussi différentes entre les stimulations gravitropiques d'un environnement normal et d'un environnement en lumière isotrope.

Le seul motif mis en évidence par l'ACP est « RGWCGTG » appelé dans la littérature DPE (Butlere et Kadonaga, 2016). Il est caractérisé comme un motif régulateur qui participerait à l'initiation de la transcription. Le promoteur minimal est le siège de l'initiation de la transcription, et il semble peu probable qu'un facteur de transcription, autre que l'ARN polymérase II, ait la place de s'y fixer. L'encombrement du promoteur minimal est dû à la présence de l'ARN polymérase II, et à l'heure actuelle, aucun des motifs du promoteur minimal n'a montré un autre rôle à part dans l'initiation de la transcription.

L'analyse d'une corrélation entre des motifs régulateurs et des résultats d'expression a été réalisée à partir d'un échantillon de l'ensemble des gènes *PtWAKs*. Cependant, les nombreux résultats d'expression manquants influencent l'analyse de l'ACP. De même, un faible nombre d'observation rend l'ACP sensible aux valeurs aberrantes. Enfin, la restriction du nombre de gènes réduit la puissance statistique de l'ACP et ne permet pas de conclure sur une relation entre la présence d'un motif et la régulation des gènes *PtWAKs*.

5.3 La relation entre les motifs et le stress hydrique

Les jeux de données concernant le stress hydrique ont été obtenus par des méthodes différentes : qPCR en temps réel (« turgescence foliaire » cf 3.4.4 qPCR de turgescence foliaire) et RNAseq (« stress hydrique » cf 3.4.5 RNAseq de stress hydrique). Par conséquent, ces jeux de données ont été traité séparément pour les analyses des corrélations. Le motif AWTTCAAA est retrouvé corrélé au stress hydrique lors de l'étude « turgescence foliaire ». Comme précisé précédemment, cette corrélation peut être due à la faible population utilisée (19 *PtWAKs*). Un alignement des séquences régulatrice des 3 *PtWAKs* (76, 84 et 89) corrélées aux variables « stress » et « présence du motif

AWTTCAAA », permettrait d'initier une étude sur l'importance de la localisation de motifs potentiellement régulateurs. Aucune corrélation entre les motifs régulateurs et les données RNAseq de stress hydrique n'a pu être mise en évidence. Ce résultat est peut être dû à un problème de format entre le jeu de données et les tests réalisés. Il existe des motifs liés au stress hydrique (Kaur et al., 2013) qui ne ressortent pas dans cette étude. Toutefois les données RNAseq ont permis de différencier les différentes *PtWAKs* en fonction de leurs niveaux d'expressions. Dans l'ACP des *PtWAKs* surexprimés forment un groupe un peu excentré par rapport au groups des « sous éxprimés » et « non différentiel », ce qui nous permet de supposer qu'il existe probablement des motifs impliqués dans cette tendance.

6. Conclusion et Perspectives

La méthode d'étude exploratoire sur une famille multigénique représentée par un très grand nombre de membres a permis de mettre en évidence l'intérêt et la nécessité de l'utilisation de la bio-informatique dans le traitement et l'utilisation de flux de données volumineux.

Dans ce jeux de résultats préliminaires obtenus, seul le motif AWTTCAAA se démarque statistiquement (par rapport à la qualité de représentation et à sa contribution) et montre une corrélation positive avec l'expérience menée sur la modulation de l'état hydrique des feuilles (cf 3.4.4 qPCR de « turgescence foliaire »). C'est le résultat qui supporte l'idée d'un lien entre un motif régulateur et la régulation des *PtWAKs*.

Il est nécessaire de perfectionner l'analyse de certaines *PtWAKs* spécifiques, particulièrement retrouvées dans les différents résultats obtenus. On peut citer dans ce cas : *PtWAK155b*, *PtWAK70* ou *PtWAK44*. L'aide d'outils en ligne comme PlantCARE (Lescot et al., 2002) (http://bioinformatics.psb.ugent.be/webtools/plantcare/html/) qui recherchent des motifs dans des bases de données dédiées, serait idéale pour mener cette étude. D'autres outils comme PLANTPAN (Chow et al., 2015) (http://plantpan2.itps.ncku.edu.tw/), TRANSFAC (Matys et al., 2003) (http://www.gene-regulation.com/pub/databases.html) et RED2 (Lajoie et al., 2012) (http://www.atgc-montpellier.fr/RED2/) seront également utiles pour approfondir cette analyse *in silico*.

L'un des problèmes majeurs rencontré est le manque d'amorces conçues pour l'amplification des

transcrits des gène *PtWAKs*. En effet, sur les 175 membres de la famille, nous en disposions d'un peu moins que la moitié. Ce problème pourrait remettre en jeu tout les résultats obtenus car les corrélations observées sont différentes si l'on étudie un population composée 40 *PtWAKs* et une autre population composée de 175 *PtWAKs*. La différence entre ces deux populations entraîne une perte d'information importante pour la compréhension et l'interprétation des résultats obtenus.

Pour conclure, ce stage m'a permis de comprendre que le travail d'équipe est essentiel pour mener à bien un sujet tel que celui-ci. C'est une coopération entre le biologiste qui pense à des expériences pour mettre en évidence des plans expérimentaux pour en comprendre les aboutissants et du bio-informaticien qui traite les données, parfois incompréhensibles, en résultats. Les perspectives envisageables pour cette étude de promoteurs chez *PtWAKs* sont tout d'abord de concevoir les amorces pour tous les membres de la famille et ainsi pouvoir suivre le comportement de chaque membre aux différents stress utilisés dans cette étude. Reconstruire un jeu d'expression complet permettra probablement d'obtenir des résultats statistiquement justifiables. Contrairement à cette étude, le jeu de données doit être uniforme car obtenir des jeux de données de différents organes, de différentes méthodes ou de différentes espèces complexifie l'étude. Après avoir trouvé un motifs intéressant, on pourra réaliser des expériences avec des mutants du motifs, ou une étude avec différentes parties du promoteur tronqué afin de valider la fonction des motifs trouvés.

Barrett, L.W., Fletcher, S., Wilton, S.D. (2012). Regulation of eukaryotic gene expression by the untranslated gene regions and other non-coding elements. Cellular and Molecular life Sciences 69, 3613-3634.
Bradnam, K.R., Korf, I. (2008). Longer first introns are a general property of eukaryotic gene structure. PLoS ONE 8, e3093.
Brutus, A., Sicilia, F., Macone, A., Cervone, F., De Lorenzo, G. (2010). A domain swap approach reveals a role of the plant wall-associated kinase 1 (WAK1) as a receptor of oligogalacturonides. Proceedings of the National Academy of Sciences 107, 9452-9457.
Butler, J.E.F., Kadonaga, J.T. (2016) The RNA polymerase II core promoter : a key component in the regulation of gene expression.Genes & Development 16, 2583-2592.
Chow, C-N., Zheng, H-Q., Wu, N-Y., Chien, C-H., Huang, H-D., Lee, T-Y., Chiang-Hsieh, Y-F., Hou, P-F., Yang, T-Y., Chang, W-C. (2016). PlantPAN 2.0 : an update of plant promoter analysis navigator for reconstructing transcriptional regulatory networks in plants. Nucleic Acids Research 44, 1154-1160.
Danimo, Y.M., Even, E., Ideses, D., Juven-Gershon, T. (2015). The core promoteur : At the heart of gene expression. Biochimica et Biophysica Acta, 1116-1131
De Oliveira, L.F.V., Christoff, A.P., de Lima, J.C., de Ross, B.C.F., Sachetto-Martins, G., Margis-Pinheiro, M., Margis, R. (2014). The Wall-associated Kinase gene family in rice genomes. Plant Science 229, 181-192.
Decreux, A., Messiaen, J. (2005). Wall-associated Kinase WAK1 Interacts with Cell Wall Pectins in a Calcium-induced Conformation. Plant and Cell Physiology 46, 268-278.
Decreux, A., Thomas, A., Spies, B., Brasseur, R., Cutsem, P.V., Messiaen, J. (2006). In vitro characterization of the homogalacturonan-binding domain of the wall-associated kinase WAK1 using site-directed mutagenesis. Phytochemistry 67, 1068-1079.
Delteil, A., Gobbato, E., Cayrol, B., Estevan, J., Michel-Romiti, C., Dievart, A., Kroj, T., Morel, J.-B. (2016). Several wall-associated kinases participate positively and negatively in basal defense against rice blast fungus. BCM Plant Biology, 16-17.
Finn, R.D., Coggill, P., Eberthardt, R.Y., Eddy, S.R., Mistry, J., Mitchell, A.L., Potter, S.C., Punta, M., Qureshi, M., Sangrador-Vegs, A., Salazar, G.A., Tate, J., Bateman, A. (2016). Nuceic Acids Research, Database Issue 44 : D279-D285.
Garcia-Hernandez, C. M., Finer, J. J. (2014). Identification and validation of promoters and *cis*-acting regulatory elements. Plant Science, 109-119.
He, Z.-H., Fujiki, M., Kohorn, B.D. (1996). A Cell Wall-associated, Receptor-like Protein Kinase. Journal of Biological Chemistry 271, 19789-19793.
He, Z.-H., He, D., Kohorn, B.D. (1998). Requirement for the induced expression of a cell wall associated receptor kinase for survival during the pathogen response. The Plant Journal 14, 55-63.
He, Z.-H., Cheeseman, I., He, D., Kohorn, B. (1999). A cluster of five cell wall-associated receptor kinase genes, Wak1–5, are expressed in specific organs of Arabidopsis. Plant Mol Biol 39, 1189-1196.
Higo, K., Ugawa, Y., Iwamoto, M., Korenaga, T. (1999). Plant cis-acting regulatory DNA elements (PLACE) database : 1999. Nucleic Acids Research 27, 297-300.
Hou, X., Tong, H., Selby, J., DeWitt, J., Peng, X., He, Z.-H. (2005). Involvement of a Cell WallAssociated Kinase, WAKL4, in Arabidopsis Mineral Responses. Plant Physiology 139, 1704-1716.
Humphrey, T. V., Bonetta, D. T., Goring, D. R. (2007). Sentinels at the wall : cell wall receptors and sensors. New Phytologist 176, 7-21.
Hu, W., Lv, Y., Lei, W., Li, X., Chen, Y., Zheng, L., Xia, Y., Shen, Z. (2014). Cloning and characterization of the Oryza sativa wall-associated kinase gene OsWAK11 and its transcriptional

response to abiotic stresses. Plant Soil 384, 335-346.
Jeffares, D.C., Mourier, T., Penny, D. (2006). The biology of intron gain and loss. Trends in Genetics 22, 16-21.
Kaur, R., Singh, K., Singh, J. (2013). A root-specific wall-associated kinase gene, HvWAK1, regulates root growth and is highly divergent in barley and other cereals. Funct. Integr. Genomics 13, 167-177.
Kohorn, B D. (2001). WAKs; cell wall associated kinases. Cell Biology, 529-533.
Kohorn, B.D., Kobayashi, M., Johansen, S., Riese, J., Huang, L.-F., Koch, K., Fu, S., Dotson, A., Byers, N. (2006). An Arabidopsis cell wall-associated kinase required for invertase activity and cell growth. The Plant Journal 46, 307-316.
Kohorn, B.D., Johansen, S., Shishido, A., Todorova, T., Martinez, R., Defeo, E., Obregon, P. (2009). Pectin activation of MAP kinase and gene expression is WAK2 dependent. The Plant Journal 60, 974-982.
Kohorn, B.D., Kohorn, S.L. (2012a). The Cell Wall Associated Kinases, WAKs, As Pectin Receptors. Frontiers in Plant Science 3.
Kohorn, B.D., Kohorn, S.L., Todorova, T., Baptiste, G., Stansky, K., McCullough, M. (2012b). A Dominant Allele of Arabidopsis Pectin-Binding Wall-Associated Kinase Induces a Stress Response Suppressed by MPK6 but Not MPK3 Mutations. Molecular Plant 5, 841-851.
Lajoie, M., Gascuel, O., Lefort, V., and Bréhélin, L. 2012. Computational discovery of regulatory elements in a continuous expression space. Genome Biology 13 : R109.
Lally, D., Ingmire, P., Tong, H.-Y., He, Z.-H. (2001). Antisense Expression of a Cell Wall–Associated Protein Kinase, WAK4, Inhibits Cell Elongation and Alters Morphology. The Plant Cell 13, 1317-1332.
Lebel, E., Heifetz, P., Thorne, L., Uknes, S., Ryals, S., Ward, E. (1998). Functional analysis of regulatory sequences controlling PR-1 gene expression in Arabidopsis. The Plant Journal 16, 223-233.
Lescot, M., Déhais, P., Thijs, G., Marchal, K., Moreau, Y., Van de Peer, Y., Rouzé, P., Rombauts, S. (2002). PlantCARE, a database of plant cis-acting regulatory elements and a portal to tools for *in silico* analysis of promoter sequences 30, 325-327.
Liao, L., Ning, G., Liu, C., Zhang, W., Bao, M. (2013). The intron from the 5'-UTR of the *FBP11* gene in petunia displays promoter-and enhancer-like functions. Scientia Horticulturae 154, 96-101.
Matys, V., Fricke1, E., Geffers, R., Göbling, E., Haubrock, M., Hehl, R., Hornischer, K., Karas, D., Kel, A.E., Kel-Margoulis, O.V., Kloos, D.-U., Land, S., Lewicki-Potapov, B., Michael, H., Münch, R., Reuter, I., Rotert, S., Saxel, H., Scheer, M., Thiele, S., Wingender, E. (2003). TRANSFAC® : transcriptional regulation, from patterns to profils. Nucleic Acids Research 31, 374-378.
Meier, S., Ruzvidzo, O., Morse, M., Donaldson, L., Kwezi, L., Gehring, C. (2010). The *Arabidopsis Wall Associated Kinase-Like 10* Gene Encodes a Functional Guanylyl Cyclase and Is CoExpressed with Pathogen Defense Related Genes. PLoS ONE 5, e8904.
Molina,C., Grotewold, E. (2005). Genome wide analysis of *Arabidopsis* core promoters. BMC Genomics, 6-25.
Park, A.R., Cho, S.K., Yun, U.J., Jin, M.Y., Lee, S.H., Sachetto-Martins, G., Park, O.K. (2001). Interaction of the Arabidopsis Receptor Protein Kinase Wak1 with a Glycine-rich Protein, AtGRP-3. Journal of Biological Chemistry 276, 26688-26693.
Parra, G., Bradnam, K., Rose, A.B., Korf, I. (2011) Comparative and functional analysis of intron-mediated enhancement signals reveals conserved features among plants. Nucleic Acids Research 39, 5328-5337.
Roy, A.L., Singer, D.S. (2015). Core promoters in transcription : old problem, new insights. Trends in Biochemical Sciences 40, 165-171.
Shen, Z., Yao, J., Sun, J., Chang, L., Wang, S., Ding, M., Qian, Z., Zhang H., Zhao, N., Sa, G., Hou, P., Lang, T., Wang, F., Zhao, R., Shen, X., Chen,S. (2015). *Populus euphratica* HSF binds the

promoter of WRKY1 to enhance salt tolerance. Plant Science 235, 89-100.
Sivaguru, M., Ezaki, B., He, Z.-H., Tong, H., Osawa, H., Baluška, F., Volkmann, D., Matsumoto, H. (2003). Aluminum-Induced Gene Expression and Protein Localization of a Cell Wall-Associated Receptor Kinase in Arabidopsis. Plant Physiology 132, 2256-2266.
Stenflo, J., Stenberg, Y., Muranyi, A. (2000). Calcium-binding EGF-like modules in coagulation proteinases: function of the calcium ion in module interactions. Biochimica et Biophysica Acta (BBA). Protein Structure and Molecular Enzymology 1477, 51-63.
Tamar, J.G., Hsu, J.-Y., Theisen, J.W.M., Kadonaga, T.J., (2008). The RNA polymerase II core promoter – the gateway to transcription. Cell Biology 20, 253-259.
Tocquard, K., Lafon-Placette, C., Augin, D., Muries, B., Bronner, G., Lopez, D., Fumanal, B., Franchel, J., Bourgerie, S., Maury, S., Label, P., Julien, J.L., Roeckel-Drevet, P., Venisse, J.S. (2014). In silico study of wall-associated kinase family reveals large-scale genomic expansion potentially connected with functional diversification in *Populus*. Tree Genetics & Genomes 10, 1135-1147.
Verica, J.A., Chae, L., Tong, H., Ingmire, P., He, Z.-H. (2003). Tissue-Specific and Developmentally Regulated Expression of a Cluster of Tandemly Arrayed Cell Wall-Associated Kinase-Like Kinase Genes in Arabidopsis. Plant Physiology 133, 1732-1746.
Verica, J.A., and He, Z. (2002). The Cell Wall-Associated Kinase (WAK) and WAK-Like Kinase Gene Family. Plant Physiology 129, 455-459.
Wagner, T.A., Kohorn, B.D. (2001). Wall-Associated Kinases Are Expressed throughout Plant Development and Are Required for Cell Expansion. The Plant Cell 13, 303-318.
Winzell, A., Aspeborg, H., Wang, Y., Ezcurra, I. (2010). Conserved CA-rich motifs in gene promoters of PtxtMYB021-responsive secondary cell wall carbohydrate-active enzymes in *Populus*. Biochemical and Biophysical Research Communications 394, 848-853.
Xiao, G., Zhang, Z.Q., Yin, C.F., Lui, R.Y., Wu, X.M., Tan, T.L., Chen, S.Y., Lu, C.M., Guan, C.Y. (2014). Characterization of the promoter and 5'-UTR intron of oleic acid deaturase (FAD2) gene in *Brassica napus*. Gene 545, 45-55.
Yamasaki, K., Kigawa, T., Inoue, M., Tateno, M., Yamasaki, T., Yabuki, T., Aoki, M., Seki, E., Matsuda, T., Nunokawa, E., Ishizuka, Y., Terada, T., Shirouzu, M., Osanai, T., Tanaka, A., Seki, M., Shinozaki, K., Yokoyama, S. (2004). A novel zinc-binding motif revealed by solution structures of DNA-binding domains of *Arabidopsis* SBP-family transcription factors. Journal of Molecular Biology 337, 49-63.
Zhang, S., Chen, C., Li, L., Meng, L., Singh, J., Jiang, N., Deng, X.-W., He, Z.-H., Lemaux, P.G. (2005). Evolutionary Expansion, Gene Structure, and Expression of the Rice Wall-Associated Kinase Gene Family. Plant Physiology 139, 1107-1124.

ANNEXE n°1

```
#!/usr/bin/env perl

=head1 NAME

DownloadPIAFdbPromoters.pl : Recovery of gene promoters from their ID (old or current) given on STDIN or passed through --geneList option. Outputs promoter sequences, optionnally masked, to standard output in a multifasta format.

=head1 SYNOPSIS

DownloadPIAFdbPromoters.pl [--verbose] [--masked] [--size=1000] [--geneList=filename.txt] <STDIN  >STDOUT

=head1 DESCRIPTION

Recovers gene promoters with their ID from PIAF's poplar genome database. ID can be present or old gene model ID. Outputs sequence, optionnally masked, as multifasta format.

=head1 OPTIONS

--Help|help|h, produces this help file.

--verbose[no-Verbose]|Verbose[no-verbose]|v[no-v], boolean option to print out warnings during execution. Warnings and errors are redirected to STDERR. Defaults to no-verbose (silent mode).

--masked[no-masked]|Masked[no-Masked]|m[no-m], boolean option to output masked or unmasked sequences. Masking information is provided by chromosome genomic sequence. Defaults to no-masked, i.e. everything uppercased.

--size|Size|s=integer, cuts each gene's promoter sequence to "size" length in bp. Must be a positive and non null integer. Defaults to 1,000 bp.

--geneList|GeneList|Genelist|g=string, optional file of genes ID (old or current). If this option is not used, the geneList must be passed using standard input. Full or relative path to filename allowed.

=head1 AUTHORS

Philippe LABEL

=head1 VERSION

1.04

=head1 DATE

12/03/2014
```

```
=cut

# libraries
use warnings;
use strict;
use Pod::Usage;
use Getopt::Long;
use File::Basename;
use DBI;

# scalars
my $help;
my $verbose;
my $database        = 'genomepeuplier';
my $host            = 'piafdb.univ-bpclermont.fr';
my $login           = 'piaf';
my $password        = 'piaf';

my $dbh;
my $geneList;
my $length_promot;
my $defaultPromoterLength = 1000;
my $masked;
my $maskedStatus;
my $compt = 0;
my $queryMyGene = "     SELECT          strand_genes,
                                        start_genes,
                                        end_genes,
                                        chromosome_id_fk_genes,
                                        name_genes
                        FROM            genes
                        WHERE           gene_id_pk ~ ?
                        OR              synonyms_genes ~ ?
                        ;";


# lists

# hashs
my %genePos;        # va contenir les infos de position des genes
my %chrSeq;# va contenir les sequences des chormosomes sur lesquels on cherche des promoteurs
my %chromToDwnl; # list of chromosomes to download sequence for

# functions
sub error {
        # management of error messages and help page layout, will stop execution
        # local arguments passed:       1st, error message to output
        my $error = shift;
        my $filename = basename($0);
        pod2usage(-message => "$filename (error): $error Execution halted.", -verbose => 1,
```

```
-noperldoc => 1);
        exit(2);
}

sub warning {
        # management of warnings and execution carry on
        # local arguments passed:       1st, warning message to output
        if ($verbose) {
                my $message = shift;
                my $filename = basename($0);
                warn("$filename (info): ".$message."\n");
        }
}

sub QueryChrSeq {
        # recupère les sequences de chaque chromosome sur lesquels on a des genes
        # local arguments passed:     $chromosomesList, list of chromosomes to query sequence for
        # global variables used:      $dbh, Connection to the database PostgreSQL
        #                                     %chrSeq, hash table of chromosome sequences
with key as chromosome number
        my $chromosomesList = shift;
        my $query = "           SELECT      chromosome_id_pk,
                                            sequence_chromosomes
                                    FROM    chromosomes
                                    WHERE   chromosome_id_pk
                                    IN      ($chromosomesList)
                                    ;";
        my $version = $dbh->prepare($query);                                    #prepare la requete
sql
        $version->execute() || die "pb de selection : $DBI::errstr";  #execution de la requete sql
        while ( my ($chr, $seq) = $version->fetchrow_array ) {          # pour chacun des
chromosomes sur lesquels on a un gene dont un cherche le promoteur
                $chrSeq{$chr} = $seq;                                                   #
store sequence in chromosome hash table
        }
        $version->finish();                                                             #
fin de la requete
}

sub QueryGene {
        # recupère les positions des genes (Chromosome, sens, brin, position)
        # local arguments passed:     $gene, id du gene dans la liste
        # global variables used:      $dbh, Connecting to the database PostgreSQL
        #                                     %genePos, hash des infos de position du
gene
        #                                     %chromToDwnl, creation d'un hachage
de noms de chromosomes à récupérer
        my $gene = shift;
        my $version = $dbh->prepare($queryMyGene);      #prepare la requete sql
        $version->execute($gene,$gene) || die "pb de selection : $DBI::errstr";     #execution de la
```

requete sql

```
	my ($strand_gene, $start_gene, $end_gene, $chromosome, $name_gene) = $version-
>fetchrow_array;	# 1 seule reponse pour 1 seul Id
	if (defined($name_gene)) { # searched gene do exist in the database
		$genePos{++$compt}{'gene'} = $name_gene;
		if ($gene ne $name_gene){
			$genePos{$compt}{'geneold'} = $gene ;
		}
		else {
			$genePos{$compt}{'geneold'} = "most recent";
		}
		$genePos{$compt}{'strand'} = $strand_gene;
		$genePos{$compt}{'start'} = $start_gene;
		$genePos{$compt}{'end'} = $end_gene;
		$genePos{$compt}{'Chr'} = $chromosome;
		$chromToDwnl{$chromosome} = "OK";	# ajout du chromosome a la liste de ceux
dont on va recuperer la sequence
	}
	else { # searched gene cannot be found in the database
		&warning("'$gene' not found in '$database'.");
	}
	$version->finish();	# fin de requete

}

MAIN: {
	GetOptions("help|Help|h"				=> \$help,
			"verbose|Verbose|v!"			=> \$verbose,
			"geneList|GeneList|Genelist|g=s"	=> \$geneList,
			"size|Size|s=i"				=> \$length_promot,
			"masked|Masked|m!"			=> \$masked
			);
	if ($help) {
		pod2usage(-verbose => 2, -noperldoc => 1);
	exit;
	}
	if (!$length_promot) {
		&warning("Using $defaultPromoterLength bp as default value for promoter size.");
		$length_promot = $defaultPromoterLength;
	}
	elsif ($length_promot <= 0){
		&error("Option Error: Size must be a positive and non null integer!");
	}
	else {
		&warning("Using $length_promot bp for promoter size.");
	}
	if ($masked) {
		&warning("Promoter sequences will be masked.");
		$maskedStatus = "masked and";
```

```
	}
	else {
		&warning("Promoter sequences will be unmasked.");
		$maskedStatus = "unmasked and";
	}
	# Connexion a la base de donnees PostGreSQL
	&warning("Connecting to database '$database' with '$login' username.");
	$dbh = DBI->connect( "dbi:Pg:dbname=$database;host=$host;", $login, $password ) or
&error("Connexion impossible a la base de donnees $database !");
	if ($geneList) {	# si on passe le fichier en argument
		&warning("Accessing external file '$geneList'.");
		open (INPUT,$geneList) or&error("Unable to read '$geneList' from '--geneList'
option !");
		while (<INPUT>){
			chomp;
			next if /^\s*$/; # skip if empty line
			&warning("Fetching gene model data for $_ on '$database'.");
			&QueryGene($_);
		}
		close INPUT;
	}
	else {
		&warning("Reading gene IDs from standard input.");
		while (<STDIN>){
			chomp;
			next if /^\s*$/; # skip if empty line
			&warning("Fetching gene model data for $_ on '$database'.");
			&QueryGene($_);
		}
	}
	&warning("Fetching required chromosome sequences.on '$database'.");
	&QueryChrSeq( """.join("', '",sort(keys(%chromToDwnl))).""");
	&warning("Outputting results to standard output.");
	my $startPos;
	my $endPos;
	my $geneName;
	my $oldGeneName;
	my $geneChromosome;
	my $seqType;
	my $strandName;
	my $lengthPromoter;
	foreach my $num (keys(%genePos)){
		$lengthPromoter = $length_promot;
		my $seq_promot = "";
		$geneName = $genePos{$num}{'gene'};
		$oldGeneName = $genePos{$num}{'geneold'};
		$geneChromosome = $genePos{$num}{'Chr'};
		if ($genePos{$num}{'strand'} eq '+'){ # working on Watson strand
			$startPos = $genePos{$num}{'start'} - $lengthPromoter - 1;
			$endPos = $genePos{$num}{'start'} - 1;
```

```
                        $seqType = "straightforward";
                        $strandName = "+ (Watson)";
                        if ($startPos < 0){      # in case there is not enough bp available for a
promoter ahead of the gene model on the chromosome
                                $startPos = 0;
                                $lengthPromoter = $endPos - $startPos;
                        }
                        $seq_promot = substr($chrSeq{$geneChromosome}, $startPos,
$lengthPromoter);
                }
                else { # working on Crick strand, so reversing start and end positions
                        $startPos = $genePos{$num}{'end'} + 1;
                        $endPos = $startPos + $lengthPromoter;
                        $seqType = "reverse-complemented";
                        $strandName = "- (Crick)";
                        # on inverse la sequence et on la complemente :
                        $seq_promot = reverse(substr($chrSeq{$geneChromosome}, $startPos,
$lengthPromoter));
                        $seq_promot =~ tr/ACGTacgt/TGCAtgca/;
                }
                        print STDOUT          ">".$geneName.
                                                " (".$oldGeneName.
                                                ") on ".$geneChromosome.
                                                " ".$strandName.
                                                " promoter ".$length_promot.
                                                " bp from ".$startPos.
                                                " to ".$endPos.
                                                " as ".$maskedStatus.
                                                " ".$seqType.
                                                "\n";
                $seq_promot = uc($seq_promot) if (!$masked); # unmask by default
                print STDOUT $seq_promot."\n"; # outputs extracted promoter sequence
        }
        &warning("Disconnecting from database '$database'.");
        $dbh->disconnect();   # deconnexion de la base de donnees

}       # end of MAIN
```

ANNEXE n°2

```
#!/usr/bin/env perl

=head1 NAME

getSeqFromPIAFdb

=head1 SYNOPSIS

getSeqFromPIAFdb [--verbose] [--masked] [--type=transcript] [--geneList=filename.txt]  <STDIN
>STDOUT

=head1 DESCRIPTION

Recovers sequences from PIAFdb poplar genome database, optionnally subtyped, from their gene
model ID (only current) given on STDIN or passed through --geneList option. Outputs sequences,
optionnally masked, to standard output in a linearized multifasta format.

=head1 OPTIONS

--Help|help|h, produces this help file.

--verbose[no-Verbose]|Verbose[no-verbose]|v[no-v], boolean option to print out warnings during
execution. Warnings and errors are redirected to STDERR. Defaults to no-verbose (silent mode).

--masked[no-masked]|Masked[no-Masked]|m[no-m], boolean option to output masked or unmasked
sequences. Masking information is provided by chromosome genomic sequence. Defaults to no-
masked, i.e. everything uppercased.

--type|Type|t=string, optional type of sequence to get from PIAFdb. Possible values are 'gene',
'transcript', 'cds', 'exon', 'intron', 'utr', '5-utr', '3-utr' and 'protein'. Defaults to 'transcript'.

--geneList|GeneList|Genelist|g=string, optional file of genes ID (only current). If this option is not
used, the geneList must be passed using standard input. Full or relative path to filename allowed.
Possible special value is 'all' when retrieving all gene models sequences.

=head1 AUTHORS

Philippe LABEL

=head1 VERSION

1.01

=head1 DATE

10/03/2015
```

```
=cut

# libraries
use warnings;
use strict;
use Pod::Usage;
use Getopt::Long;
use File::Basename;
use DBI;

# scalars
my $help;
my $verbose;
my $database		= 'genomepeuplier';
my $host		= 'piafdb.univ-bpclermont.fr';
my $login		= 'piaf';
my $password		= 'piaf';

my $dbh;
my $geneList;
my $sequenceType;
my $masked;
my $maskedStatus;
my $compt = 0;
my $query;

# lists

# hashs
my %genePos;		# va contenir les infos de position des genes
my %chrSeq;# va contenir les sequences des chormosomes sur lesquels on cherche des promoteurs
my %chromToDwnl; # list of chromosomes to download sequence for

# functions
sub error ($) {
	# management of error messages and help page layout, will stop execution
	# local arguments passed:	1st, error message to output
	my $error = shift;
	my $filename = basename($0);
	pod2usage(-message => "$filename (error): $error Execution halted.", -verbose => 1,
-noperldoc => 1);
	exit(2);
}

sub warning ($) {
	# management of warnings and execution carry on
	# local arguments passed:	1st, warning message to output
	if ($verbose) {
		my $message = shift;
		my $filename = basename($0);
		warn("$filename (info): ".$message."\n");
	}
```

```
}

sub fetchOutputSequence ($$) {
	my $geneName = shift;
	my $seqType = shift;
	warning("Fetching ".$geneName." gene model.");
	my $GeneQuery = buildQuery($seqType,$geneName);
	outputQueryGene($geneName,$GeneQuery,$seqType);
}

sub outputQueryGene ($$$) {
	my $gene = shift;
	my $query = shift;
	my $type = shift;
	my $version = $dbh->prepare($query);	#prepare la requete sql
	$version->execute() || die "pb de selection : $DBI::errstr";  #execution de la requete sql
		my $refList = $version->fetchall_arrayref;
		$version->finish();	# fin de requete
		if ($#{$refList} != -1) {
			warning ("Producing ".($#{$refList} + 1)." relevant sequence(s) for
'$gene'.");
			foreach my $refGene (@{$refList}) {
				outputSequenceFasta($$refGene[0],$$refGene[1],$$refGene[2]);
			}
		}
		else {
			warning("Sequence of type '".$type."' from gene model named '".$gene."' not
found in '$database'.");
		}
}

sub outputSequenceFasta ($$$) {
	my $geneID = shift;
	my $sequence = shift;
	my $type = shift;
	$sequence = uc($sequence) if (!$masked);
	print STDOUT ">".$geneID." ".$maskedStatus." from ".$type."\n".$sequence."\n";
}

sub buildQuery ($$) {
	my $seqType = shift;
	my $geneName = shift;
	my $query;
	if ($seqType eq "five_prime_UTR" or $seqType eq "three_prime_UTR" or $seqType eq
"fullCDS" or $seqType eq "intron" or $seqType eq "exon" or $seqType eq "protein") {
		$query = "	SELECT		feature_id_pk,
						sequence_features,
						name_genes
				FROM		genes,
						messagers,
```

```
                                        gene_features
                                WHERE   messager_id_fk_features =
messager_id_pk
                                AND     gene_id_fk_messagers = gene_id_pk
                                AND     type_features = '$seqType'
                                AND     name_genes = '$geneName'
                                ;";
        }
        elsif ($seqType eq "gene"){
                $query = "      SELECT          gene_id_pk,
                                        sequence_genes,
                                        name_genes
                                FROM            genes
                                WHERE           name_genes = '$geneName'
                                ;";
        }
        elsif ($seqType eq "transcript"){
                $query = "      SELECT          messager_id_pk,
                                        sequence_messagers,
                                        name_genes
                                FROM            genes,
                                        messagers
                                WHERE           gene_id_fk_messagers = gene_id_pk
                                AND     name_genes = '$geneName'
                                ;";
        }
        elsif ($seqType eq "utr"){
                $query = "      SELECT          feature_id_pk,
                                        sequence_features,
                                        name_genes
                                FROM            genes,
                                        messagers,
                                        gene_features
                                WHERE           messager_id_fk_features =
messager_id_pk
                                AND     gene_id_fk_messagers = gene_id_pk
                                AND     name_genes = '$geneName'
                                AND     type_features like '%_prime_UTR%'
                                ;";
        }
        else {
                error("Undefined sequence type '$seqType' to build the query.");
        }
        return($query);
}

MAIN: {
        GetOptions("help|Help|h"                                => \$help,
                        "verbose|Verbose|v!"                    => \$verbose,
                        "geneList|GeneList|Genelist|g=s"    => \$geneList,
                        "type|Type|t=s"                                 => \
$sequenceType,
```

```
                        "masked|Masked|m!"                      => \$masked
                        );
if ($help) {
        pod2usage(-verbose => 2, -noperldoc => 1);
exit;
}
if ($sequenceType) { # equivalencing option values and actual field values in the PIAFdb
        if ($sequenceType eq "cds") {
                $sequenceType = "fullCDS";
        }
        elsif($sequenceType eq "transcript") {
                $sequenceType = "transcript";
        }
        elsif($sequenceType eq "exon") {
                $sequenceType = "exon";
        }
        elsif($sequenceType eq "5-utr") {
                $sequenceType = "five_prime_UTR";
        }
        elsif($sequenceType eq "3-utr") {
                $sequenceType = "three_prime_UTR";
        }
        elsif($sequenceType eq "gene") {
                $sequenceType = "gene";
        }
        elsif($sequenceType eq "intron") {
                $sequenceType = "intron";
        }
        elsif($sequenceType eq "utr") {
                $sequenceType = "utr";
        }
        elsif($sequenceType eq "protein") {
                $sequenceType = "protein";
        }
        else {
                error("Unknown '$sequenceType' sequence type given in option.");
        }
        warning("Using '$sequenceType' sequence type.");
}
else {
        $sequenceType = "transcript";
        warning("Using default '$sequenceType' sequence type.");
}
if ($masked) {
        warning("Sequences will be masked.");
        $maskedStatus = "masked";
}
else {
        warning("Sequences will be unmasked.");
        $maskedStatus = "unmasked";
```

```
	}
	# Connexion a la base de donnees PostGreSQL
	warning("Connecting to database '$database' with '$login' username.");
	$dbh = DBI->connect( "dbi:Pg:dbname=$database;host=$host;", $login, $password ) or
&error("Connexion impossible a la base de donnees $database !");
	# managing the two modes of data input, either by file or by standard input
	if ($geneList && $geneList ne "all") {		# si on passe le fichier en argument -> file input
		warning("Accessing external file '$geneList'.");
		open (INPUT,$geneList) or&error("Unable to read '$geneList' from '--geneList'
option !");
		while (<INPUT>){
			chomp;
			next if /^\s*$/; # skip if empty line
			fetchOutputSequence($_,$sequenceType);
		}
		close INPUT;
	}
	elsif ($geneList && $geneList eq "all") { # specific case where file input specifies 'all' genes
		# récupérer la liste de tous les gènes, puis lancer la requête gène par gène
		warning("Fetching ALL gene models.");
		$query = "	SELECT		name_genes
				FROM		genes
				;";
		my $version = $dbh->prepare($query);
		$version->execute() || die "pb de selection : $DBI::errstr";
		my $refGenesList = $version->fetchall_arrayref;
		$version->finish();
		foreach my $refGene (@{$refGenesList}) { # requete gène par gène
			fetchOutputSequence($$refGene[0],$sequenceType);
		};
	}
	else { # or standard input mode
		warning("Reading gene IDs from standard input.");
		while (<STDIN>){
			chomp;
			next if /^\s*$/; # skip if empty line
			fetchOutputSequence($_,$sequenceType);
		}
	}
	warning("Disconnecting from database '$database'.");
	$dbh->disconnect();	# deconnexion de la base de donnees

}	# end of MAIN
```

ANNEXE n°3

```
#!/usr/bin/env perl

=head1 NAME

detectMotifs

=head1 SYNOPSIS

detectMotifs [--verbose] [--header] [--motifsFile=filename.fasta] [--removeDenovo=filename.fasta]
[--minLength=4] [--maxLength=33] [complexity=0] [--limit=1] [--keepHomopolymers] <STDIN
>STDOUT

=head1 DESCRIPTION

Detects motifs in sequences provided as (multi)fasta on STDIN. If a motif list is provided with
'motifsFile' option, detectMotifs will perform the detection in the sequences provided on STDIN
from this motif list. In that case, the name of the searched motif is extracted from accession name of
the fasta sequence given for the motifs list. 'minLength' and 'maxLength' options do not apply in
this case. Degenerated motifs patterns are allowed. Be careful, it can be combinatorially NP hard to
solve. To help prevent this problem, there is an optional complexity level to filter out potentially
unspecific motifs. If the motifs list is not provided, then detectMotifs will perform a de novo motif
search based only on k-mer size options values given by 'minLength' and 'maxLength'. In that case
the motif name will be numbered arbitrarily from 1 prefixed with 'motif'. Homopolymers detected
in motifs can optionally be removed from analysis (see below). A list of predetermined motifs can
also be removed from the list of motifs de novo detected. Output is made on STDOUT with a TAB-
separated format where columns are ordered as: MotifID, MotifSequence, TargetID, MotifPosList,
Counts.

=head1 OPTIONS

--Help|help|h, produces this help file.

--verbose[no-verbose]|Verbose[no-Verbose]|v[no-v], boolean option to print out warnings during
execution. Warnings and errors are redirected to STDERR. Defaults to no-verbose (silent mode).

--header|Header|d, boolean option to output a header describing data columns. Defaults to no-
header output.

--motifsFile|MotifsFile|m="filename.fasta", optional motif filename containing motifs to search for.
Should be formatted in (multi)fasta. If not provided, detectMotifs will do a blind search based on
size of motifs based on 'minLength' and 'maxLength' optionnally given.

--removeDenovo|RemoveDenovo|r="filename.fasta", optional list of motifs to be removed from
results of de novo motif detection. This file should be formatted in (multi)fasta. Defaults to empty
list. Using this option along with '--motifsFile' option is inefficient since it will consume useless
CPU time. You are then better to remove this list explicitely from your searched list and avoid use
this option.
```

--minLength|MinLength|min=4, optional value of minimal motif length to detect de novo in sequences provided on STDIN. Defaults to 4 nucleotides. Minimum of 2 nucleotides. Useful only in the case of de novo detection, detectMotifs will then mask target sequences including chunks of N shorter than MinLength.

--maxLength|MaxLength|max=33, optional value of maximal motif length to detect de novo in sequences provided on STDIN. Defaults to 33 nucleotides. Useful only in the case of de novo detection, detectMotifs will then mask target sequences including chunks of N longer than MaxLength.

--complexity|Complexity|c=0, optional complexity value (in percent) to enable complexity filtering on motif. Defaults to no complexity filtering (using 0% value). Motifs with a complexity strictly less than this value will be skipped. This value is calculated from LZW compressibility on the motif length.

--limit|Limit|l=1, optional limit of how many de novo most frequent motifs you want on output per motif length for each target sequence. Defaults to 1, i.e. outputs only the de novo most frequent per motif length for each target sequence. This option is useless when using option '--motifsFile="filename"' since if you provide a list of searched motifs, you'll want all the results for them... unless you don't really understand what happens with de novo motif search :). An error is raised if you use both options at the same time.

--keepHomopolymers|KeepHomopolymers|k, optional boolean to keep any homopolymer in motif list. Defaults to removing homopolymers. Homopolymers are polyA, polyT, polyG and polyC. PolyN are always removed whatever this option because they hold (almost) no information.

=head1 AUTHORS

Philippe LABEL

=head1 VERSION

1.01

=head1 DATE

15/02/2016

=cut

```
# libraries
use warnings;
use strict;
use Pod::Usage;
use Getopt::Long;
use File::Basename;
use Data::Dumper;
use Bio::SeqIO;
use Compress::LZF;
use Bio::PrimarySeq;
use Bio::Tools::IUPAC;
```

```
use 5.10.0;

# scalars
my $help;
my $verbose;
my $debug;                                  # debug purposes only
my $motifFileName;
my $removeFilename;
my $motifId = 1;
my $minLength;
my $maxLength;
my $complexity;
my $header;
my $limit;
my $keepHomopolymers;

# lists
my @resultsMotif;
my @removeMotifsRegex;

# hashs
my %motifs;                                 # predetermined motifs
my %motifListTarget;                                # 2D hash, list of motifs per sequence for each
motif length
my %motifPosListTarget;                             # 2D hash, list of positions on a sequence for
each motif
my %countMotifLengthListTarget;                     # 3D hash, occurence of motifs per sequence
and per motif length
my %motifName;                                      # unique motif name
my %removeMotifs;

# functions
sub error ($) {
        # management of error messages and help page layout, will stop execution
        # local arguments passed:       1st, error message to output
        my $error = shift;
        my $filename = basename($0);
        pod2usage(-message => "$filename (error): $error Execution halted.", -verbose => 1,
-noperldoc => 1);
        exit(2);
}

sub warning ($) {
        # management of warnings and execution carry on
        # local arguments passed:       1st, warning message to output
        if ($verbose) {
                my $message = shift;
                my $filename = basename($0);
                warn("$filename (info): ".$message."\n");
        }
```

```
}

sub debug ($) {
	# management of debugging messages
	# local arguments passed:	1st, warning message to output
	# no return value
	if ($debug) {
		my $message = shift;
		warn(Dumper($message));
	}
}

sub chi_squared_test ($$) { # unused for the moment...
	my $observed = shift;
	my $expected = shift;
	my $chi_squared = ($observed - $expected)**2 / $expected;
	my $probability = chisqrprob(1, $chi_squared);
	return $probability;
}

sub list_all_positions ($$) {
	# detect all occurences of a single motif in one target sequence
	# local arguments passed	1st, target sequence
	#				2nd, motif to look for
	# return value			a table of positions of the searched motif in the target sequence
	my $targetSeq = shift;
	my $motif = shift;
	my @result;
	$_ = $targetSeq;
	while (/(?=($motif))/g) { # detection of overlapping motifs included
		push(@result, $-[0]);
	}
	return @result
}

sub find_IUPAC_regex ($) {
	my $sequence = shift;
	my $ambiseq = Bio::PrimarySeq->new(-seq => $sequence, -alphabet => 'dna'); # create the
degenerated dna sequence motif
	my $iupac = Bio::Tools::IUPAC->new(-seq => $ambiseq); # create all alternative motifs
	return $iupac->regexp(); # find the regex corresponding to the degenerated dna sequence
motif
}

MAIN: {
	GetOptions(	"help|Help|h"				=> \$help,
			"verbose|Verbose|v!"			=> \$verbose,
			"header|Header|d!"			=> \$header,
			"motifsFile|MotifsFile|m=s"		=> \$motifFileName,
			"removeDenovo|RemoveDenovo|r=s"			=> \
$removeFilename,
			"minLength|MinLength|min=i"			=> \$minLength,
```

```
                        "maxLength|MaxLength|max=i"                     => \$maxLength,
                        "complexity|Complexity|c=i"             => \$complexity,
                        "limit|Limit|l=i"                       => \$limit,
                        "keepHomopolymers|KeepHomopolymers|k!"          => \
$keepHomopolymers,
                        "debug!"                                => \$debug
                                );
        if ($help) {
                pod2usage(-verbose => 2, -noperldoc => 1);
        exit;
        }

        if (! $minLength) {
                $minLength = 4;
        }
        elsif( $minLength <2 ) {
                error("Minimum motif length out of range '$minLength'");
        }

        if (! $maxLength) {
                $maxLength = 33;
        }
        elsif( $maxLength <1 ) {
                error("Maximum motif length out of range '$maxLength'");
        }

        if($minLength > $maxLength) {
                ($minLength,$maxLength) = ($maxLength,$minLength);
                warning("Swapping 'minLength' and 'maxLength' because of inverted values.");
        }
        warning("Minimum motif length set to $minLength");
        warning("Maximum lotif length set to $maxLength");

        if (! $complexity) {
                $complexity = 0;
        }
        elsif ($complexity < 0) {
                error("Negative complexity is not allowed: $complexity");
        }
        elsif ($complexity > 100) {
                error("Complexity above 100% is not allowed: $complexity");
        }
        warning("Sequence complexity threshold set to $complexity %");

        if ($limit and $motifFileName) {
                error("Using both options '-limit=$limit' and '-motifsFile=$motifFileName' is weird!
Why would you?");
        }

        if (! $limit) {
```

```
		$limit = 1;
	}
	elsif( $limit <1 ) {
		error("Limit value out of range '$limit'");
	}
	warning("Limit of most frequent motif detected per target set to $limit");

	if ($keepHomopolymers) {
		warning("Homopolymers will be removed from detected motif list");
	}

	print STDOUT "MotifID\tMotifSequence\tTargetID\tMotifPosList\tCounts\n" if ($header);

	if ($removeFilename) { # using motif list to remove from de novo results of motif detection
		error("Provided filename ('$removeFilename') doesn't exist.") if (! -e
$removeFilename);

		warning("Processing motifs list to be removed from file...");
		open(MOTIFILE,$removeFilename) or error("Unable to open '$removeFilename'
file.");
		my $seqMotif = Bio::SeqIO->new(-fh => \*MOTIFILE,
					-format => "fasta",
					);
		while ( my $seq = $seqMotif->next_seq ) { # do this for each motif in the provided
file
			$removeMotifs{$seq->display_id} = [$seq->seq(), find_IUPAC_regex($seq-
>seq())];
			push (@removeMotifsRegex, (find_IUPAC_regex($seq->seq),$seq->seq)); #
store regex and original motif to remove, in case degenerated motif is searched
		}
		close MOTIFILE or error("Unable to close $removeFilename file.");
		warning("...finished");
	}

	if ($motifFileName) { # using motif list from provided multifasta file
		error("Provided filename ('$motifFileName') doesn't exist.") if (! -e
$motifFileName);

		warning("Processing motifs list to detect from file...");
		open(MOTIFILE,$motifFileName) or error("Unable to open '$motifFileName'
file.");
		my $seqMotif = Bio::SeqIO->new(-fh => \*MOTIFILE,
					-format => "fasta",
					);
		while ( my $seq = $seqMotif->next_seq ) { # do this for each motif in the provided
file
			if (((length(compress($seq->seq)) - 1) / $seq->length * 100) < $complexity) {
				warning("Motif '".$seq->display_id."' filtered out because of low
complexity");
				next;
			}
			if ($seq->seq ~~ @removeMotifsRegex) {
```

```
                                warning("Explicitely removing '".$seq->display_id."', '".
$removeMotifs{$seq->display_id}[0]."' upon user's request from search");
                                next;
                        }
                        $motifs{$seq->display_id} = [$seq->seq, find_IUPAC_regex($seq->seq)] ; #
store motif sequence and its regex
                }
                close MOTIFILE or error("Unable to close $motifFileName file.");
                warning("...finished");
                warning("Processing sequences on STDIN...");
                my $seqSTDIN = Bio::SeqIO->new(         -fh             => \*STDIN,
                                                -format         => "fasta",
                                                );
                while ( my $seq = $seqSTDIN->next_seq ) {
                        my $sequence = $seq->seq;
                        if ($sequence =~ /N{$minLength,}/) {
                                $sequence =~ s/(N{$minLength,})/lc($1)/ge; # replace long N chunks
with their lowercase counterpart
                                warning("Masking long N chunks for '".$seq->display_id."'"."\n".
$sequence);
                        }
                        foreach my $motifId (keys(%motifs)){
                                my $search = ${$motifs{$motifId}}[1]; # working with the second
term which is the regex, first term is the degenerated motif in IUPAC standard
                                if ($seq->seq =~ /$search/) {
                                        my @motifList = list_all_positions($seq->seq,$search);
                                        print STDOUT        $motifId."\t".
                                                        ${$motifs{$motifId}}[0]."\t".
                                                        $seq-
>display_id."\t(".join(";",@motifList).")\t".
                                                        ($#motifList + 1).
                                                        "\n"; # using motif IUPAC sequence (item 0)
                                }
                        }
                }
                warning("...finished");
        }
        else { # defining motif list de novo using k-mer analysis
                warning("Processing sequences on STDIN...");
                my $seqSTDIN = Bio::SeqIO->new(         -fh             => \*STDIN,
                                                -format         => "fasta",
                                                );
                while ( my $seq = $seqSTDIN->next_seq ) {
                        if ($maxLength > $seq->length) {
                                warning("Target sequence '".$seq->display_id."' escaped because
motif length to be looked for exceeds target sequence length");
                                next;
                        }
                        for (my $motifLength = $minLength ; $motifLength <= $maxLength ;
$motifLength++){ # scan all sequences for motifs enumeration
```

```
                                my @undefMotifs = ("N" x $motifLength);
                                push( @undefMotifs, ("A" x $motifLength, "T" x $motifLength, "G" x $motifLength, "C" x $motifLength)) if (! $keepHomopolymers);
                                for (my $offset = 0 ; $offset < $motifLength ; $offset++) { # shift at each nucleotide
                                        my $sequence = substr($seq->seq,$offset);
                                        next if (length($sequence) < $motifLength);
                                        my @motifs = unpack("(A$motifLength)*", $sequence); # cuts sequence into equal length pieces
                                        foreach my $motif (@motifs) { # count complete motif and avoid counting polyN or homopolymers motifs, as well as too few complex motifs
                                                if ($motif ~~ @removeMotifsRegex) {
                                                        warning("Explicitely de novo removing '". $motif."' upon user's request from search");
                                                        next;
                                                }
                                                if (length($motif) == $motifLength and ! ($motif ~~ @undefMotifs) and ! (((length(compress($motif)) - 1) / length($motif) * 100) < $complexity)) {
                                                        my @motifPosList = list_all_positions($seq->seq,$motif); # collects all motifs positions in this sequence
                                                        $motifPosListTarget{$seq->display_id}{$motif} = [@motifPosList]; # store this list for this sequence and this motif
                                                        $countMotifLengthListTarget{$seq->display_id}{$motifLength}{$motif} = $#motifPosList + 1; # store the number of occurrence of this motif in this sequence for each motif length
                                                        $motifName{$motif} = "motif".$motifId++ if (! exists($motifName{$motif})); # create, name and store the motif name only once
                                                }
                                                else {
                                                        warning("De novo motif '".$motif."' removed");
                                                }
                                        }
                                }
                        # now for each motif length we have the motifs list and their occurrences, lets take the most represented
                        my @resultsMotifs = sort({$countMotifLengthListTarget{$seq->display_id}{$motifLength}{$b} <=> $countMotifLengthListTarget{$seq->display_id}{$motifLength}{$a}} keys(%{$countMotifLengthListTarget{$seq->display_id}{$motifLength}})); # sort
                        @resultsMotifs = @resultsMotifs[0..($limit - 1)]; # prune
                        warning("Useless sequence '".$seq->display_id."' (".$seq->seq.")") if (! @resultsMotifs);
                        next if (! @resultsMotifs);
                        $motifListTarget{$seq->display_id}{$motifLength} = [@resultsMotifs]; # and store
                        }
                }
                foreach my $seqTarget (keys(%motifListTarget)) { # proceed to output of results
                        foreach my $motifLength (sort({$a <=> $b} keys(%{$motifListTarget{$seqTarget}}))) {
                                foreach my $motif (@{$motifListTarget{$seqTarget}{$motifLength}}) {
                                        next if (! $motif);
```

```
                                        print STDOUT          $motifName{$motif}."\t".
                                                              $motif."\t".
                                                              $seqTarget."\t(".
                                                              join(";",@{$motifPosListTarget{$seqTarget}
{$motif}}).")\t".
                                                              $countMotifLengthListTarget{$seqTarget}
{$motifLength}{$motif}.
                                                              "\n";
                                    }
                                }
                        }
                        warning("...finished");
                }

}       # end of MAIN
```

ANNEXE n°4

```
#!/usr/bin/env perl

=head1 NAME

analyseMotifs

=head1 SYNOPSIS

analyseMotifs [--verbose] [--header] [--sort="targetFreq-"] [--allDetails] <STDIN >STDOUT

=head1 DESCRIPTION

analyses motifs statistics provided as tab-separated dataflow on STDIN, typically obtained from
'detectMotifs' tool. Input on STDIN should be structured as TAB-separated format where columns
are ordered as: motifID, motifSeq, (motifLength), (targetIDs), (posLists), (countsLists), nbTargets,
(allTargets), targetFreq, meanMatch, varMatch, medianMatch, minMatch, maxMatch, nbMatches,
(allMatches) and matchProp. Variable names in parenthesis are outputted optionnaly with
'allDetails' option. Header can be optionnally present on input. Results will be output on STDOUT.

=head1 DEFINITIONS

targetFreq
Frequency calculated for each motif and based on the number of hits (including several times in the
same sequence (target)) for this motif compared to the total number of sequences analysed.
targetFreq = nbTargets / allTargets. It tells the frequency of finding this motif among all sequences
analyzed, aka how frequently you can meet this motif walking along all sequences. It depends on
the occurrence of this motif, but also on the amount of sequences to be analyzed. Beware that if
motif accessions are detected from different sources such as gene model name or transcript model
name, these sources will be considered as separate targets, missing the gene level.

matchProp
Proportion calculated for each motif and based on the number of matches for this motif on all
sequences (genes) compared to number of matches of all motifs on all genes.
matchProp = nbMatches / sum(nbMatches). It tells how much this motif is found among all other
motifs detected, aka how dominant this motif is among all others.

=head1 OPTIONS

--Help|help|h, produces this help file.

--verbose[no-verbose]|Verbose[no-Verbose]|v[no-v], boolean option to print out warnings during
execution. Warnings and errors are redirected to STDERR. Defaults to no-verbose (silent mode).

--header|Header|d, boolean option to indicate the presence of a header describing data columns on
input. Defaults to no-header expected on output. If this option is set, then the same header given on
STDIN will be used for output on STDOUT.
```

--sort|Sort|s="targetFreq-", optionally sort results on STDOUT by decreasing order (the highest value, the first out) on columns. Defaults to sorting the single column 'targetFreq' in descending order. Sort can combine several columns using a comma-separated list of column names in the string passed to 'sort' option. For example: -sort="motifLength+,matchProp-,targetFreq-" will sort output by increasing motif length first, then for tie, matchProp will be sorted decreasingly and if tied again targetFreq will be sorted decreasingly. The first column given, the dominant key sort. Eventually, a plus (+) or minus (-) sign can be appended to column name to indicate the sorting order: + = ascending, - = descending (default). If omitted, a descending sort will be done. Column names can be any value within motifLength, nbTargets, targetFreq, meanMatch, varMatch, medianMatch, minMatch, maxMatch, nbMatches and/or matchProp.

--allDetails|AllDetails|a, boolean option to obtain results in full details including, positions list (posLists), target ID list (targeIDs), motif length and counts lists for each motif; plus total targets analyzed and total matches overall detected. Defaults to no details.

=head1 AUTHORS

Philippe LABEL

=head1 VERSION

1.00

=head1 DATE

15/02/2016

=cut

```
# libraries
use warnings;
use strict;
use Pod::Usage;
use Getopt::Long;
use File::Basename;
use List::Util qw< sum >;
use Statistics::Distributions qw< chisqrprob >;
use Statistics::Multtest qw(:all);
use Statistics::Descriptive;
use Data::Table;
use Data::Dumper;
use 5.10.0;

# scalars
my $help;
my $verbose;
my $debug;                                  # debug purposes only
my $header;
my $sorting;
my $allDetails;
my $stats = Statistics::Descriptive::Full->new();
my $allMatchSum;
```

```
my $allTargetSum;

# lists
my @colNames = (    "motifID",
                    "motifSeq",
                    "motifLength",
                    "targetIDs",
                    "posLists",
                    "countsLists",
                    "nbTargets",
                    "allTargets",
                    "targetFreq",
                    "meanMatch",
                    "varMatch",
                    "medianMatch",
                    "minMatch",
                    "maxMatch",
                    "nbMatches",
                    "allMatches",
                    "matchProp");

my @datatypes =     (1,
                    1,
                    0,
                    1,
                    1,
                    1,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0,
                    0 );

my @detailedColumns = (     "motifLength",
                    "targetIDs",
                    "posLists",
                    "countsLists",
                    "allTargets",
                    "allMatches");

my @sortCriteria;
my @dataResults;
my @headers;
my @sortParams;
```

```
# hashs
my %accessions;
my %sequences;
my %targets;
my %positions;
my %counts;
my %countsDetails;
my %allTargets;
my %dataTypes;
@dataTypes{@colNames} = @datatypes;
my %sortOrder;
my %viewedSequences;

# functions
sub error ($) {
	# management of error messages and help page layout, will stop execution
	# local arguments passed:	1st, error message to output
	my $error = shift;
	my $filename = basename($0);
	pod2usage(-message => "$filename (error): $error Execution halted.", -verbose => 1,
-noperldoc => 1);
	exit(2);
}

sub warning ($) {
	# management of warnings and execution carry on
	# local arguments passed:	1st, warning message to output
	if ($verbose) {
		my $message = shift;
		my $filename = basename($0);
		warn("$filename (info): ".$message."\n");
	}
}

sub debug ($) {
	# management of debugging messages
	# local arguments passed:	1st, warning message to output
	# no return value
	if ($debug) {
		my $message = shift;
		warn(Dumper($message));
	}
}

sub chi_squared_test ($$) { # unused for the moment...
	my $observed = shift;
	my $expected = shift;
	my $chi_squared = ($observed - $expected)**2 / $expected;
	my $probability = chisqrprob(1, $chi_squared);
	return $probability;
}
```

```
sub find_IUPAC_regex ($) {
	my $sequence = shift;
	my $ambiseq = Bio::PrimarySeq->new(-seq => $sequence, -alphabet => 'dna'); # create the
degenerated dna sequence motif
	my $iupac = Bio::Tools::IUPAC->new(-seq => $ambiseq); # create all alternative motifs
	return $iupac->regexp(); # find the regex corresponding to the degenerated dna sequence
motif
}

MAIN: {
	GetOptions(	"help|Help|h"				=> \$help,
			"verbose|Verbose|v!"			=> \$verbose,
			"header|Header|d!"			=> \$header,
			"sort|Sort|s=s"				=> \$sorting,
			"allDetails|AllDetails|a!"		=> \$allDetails,
			"debug!"				=> \$debug
				);
	if ($help) {
		pod2usage(-verbose => 2, -noperldoc => 1);
	exit;
	}

	if ($sorting) {
		foreach my $keyCol (split(",", $sorting)) {
			my ($colName,$order) = $keyCol =~ /(.*[^+-])([+-]?)/;
			error("Unknown column name '$colName' to sort with.") if (! ($colName ~~
@colNames));
			$order = 1 if ($order ~~ ["","-"]);
			$order = 0 if ($order eq "+");
			push (@sortCriteria, $colName);
			$sortOrder{$colName} = $order;
		}
	}
	else {
		push (@sortCriteria, "targetFreq");
		$sortOrder{"targetFreq"} = 1;
	}
	warning("Sorting order set to '".join(">>",@sortCriteria)."'");

	warning("Processing motifs list from STDIN...");
	# define header composition
	foreach my $col (@colNames) {
		if ($allDetails or ! ($col ~~ @detailedColumns) or ($col ~~ @sortCriteria) ) {
			push (@headers, $col);
		}
	}
	while ( my $line = <STDIN> ) {
		chomp $line;
		my @fields = split("\t", $line);
```

```
		if (exists($viewedSequences{$fields[1]})) {
			$accessions{$viewedSequences{$fields[1]}}++;
			push(@{$sequences{$viewedSequences{$fields[1]}}}, $fields[1]);
			push(@{$targets{$viewedSequences{$fields[1]}}}, $fields[2]);
			push(@{$positions{$viewedSequences{$fields[1]}}}, $fields[3]);
			push(@{$countsDetails{$viewedSequences{$fields[1]}}}, $fields[4]);
		}
		else {
			$viewedSequences{$fields[1]} = $fields[0];
			$accessions{$fields[0]}++;
			push(@{$sequences{$fields[0]}}, $fields[1]);
			push(@{$targets{$fields[0]}}, $fields[2]);
			push(@{$positions{$fields[0]}}, $fields[3]);
			push(@{$countsDetails{$fields[0]}}, $fields[4]);
		}
		$allTargets{$fields[2]}++;
		$allMatchSum += $fields[4];
	}

	my $allTargets = scalar(keys(%allTargets)) + 1;

	foreach my $accession (keys(%accessions)) {
		my @dataTable = ();
		$stats->clear;
		$stats->add_data(sort(@{$countsDetails{$accession}}));
		push(@dataTable, $accession);                                   # accession
of the motif
		push(@dataTable, ${$sequences{$accession}}[0]);                 # sequence
of the motif
		push(@dataTable, sprintf("%d",length(${$sequences{$accession}}[0]))) if
($allDetails or ("motifLength" ~~ @sortCriteria));  # motif length in (degenerated) bases
		push(@dataTable, join(";",@{$targets{$accession}})) if ($allDetails or ("targetIDs"
~~ @sortCriteria));# list of targets for this motif
		push(@dataTable, join("_",@{$positions{$accession}})) if ($allDetails or
("posLists" ~~ @sortCriteria));# list of all match positions for this motif on all targets
		push(@dataTable, join(";",@{$countsDetails{$accession}})) if ($allDetails or
("countsLists" ~~ @sortCriteria));              # list of count matches for this motif on each target
		push(@dataTable, $accessions{$accession});                      # number
of targets for this motif
		push(@dataTable, sprintf("%d",$allTargets)) if ($allDetails or ("allTargets" ~~
@sortCriteria));        # the total of targets for all the motifs, i.e. the overall number of targets
analyzed
		push(@dataTable, sprintf("%.3f",($accessions{$accession} / $allTargets)));# the
frequency of targets for this motif relatively to all targets analyzed
		push(@dataTable, sprintf("%.2f",$stats->mean));                 # mean
match per target for this motif
		push(@dataTable, sprintf("%.2f",$stats->variance));             # variance
of mean match per target for this motif
		push(@dataTable, sprintf("%.2f",$stats->median));               # median of match
per target for this motif
		push(@dataTable, sprintf("%d",$stats->min));                    # guess
what?
```

```
                push(@dataTable, sprintf("%d",$stats->max));                    # ohhh,
surprise! the max of match per target for this motif
                push(@dataTable, sprintf("%d",$stats->sum));                    # the sum
of match on all targets for this motif
                push(@dataTable, sprintf("%d",$allMatchSum)) if ($allDetails or ("allMatches" ~~
@sortCriteria));        # the total of matches on all targets for all the motifs
                push(@dataTable, sprintf("%.3f",($stats->sum / $allMatchSum)));         # the
frequency of matches for this motif relatively to total matches on all targets for all motifs
                push(@dataResults, [@dataTable]);
        }
        warning("...finished");

        warning("Sorting results...");
        my $dataResults = Data::Table->new(\@dataResults, \@headers, 0);
        foreach my $sortKey (@sortCriteria) {
                push (@sortParams,$sortKey); # column name to sort
                push (@sortParams,$dataTypes{$sortKey}); # type numerical
                push (@sortParams,$sortOrder{$sortKey}); # sort order
        }
        push (@sortParams,'motifID'); # force final sort on motifID
        push (@sortParams,1); # type non-numeric
        push (@sortParams,0); # sort ascending
        $dataResults->sort(@sortParams);
        warning("...finished");

        warning("Outputting results on STDOUT...");
        print STDOUT $dataResults->tsv(defined($header));
        warning("...finished");

}       # end of MAIN
```

ANNEXE N°5

```
#!/usr/bin/env perl

=head1 NAME

testMotifs

=head1 SYNOPSIS

testMotifs [--verbose] [--testGroup=filename] [--refGroup=filename] [--fdr=0.01] [--header]
<STDIN >STDOUT

=head1 DESCRIPTION

Estimates significance of motifs occurrence between two lists of motifs. First list is the test group
and second list is the reference group. Both lists are provided as files through mandatory options.
'testMotifs' will compute a chi-square test between test and reference motif abundances and adjust
pvalues with a Benjamini-Yekutieli (2001, The Annals of Statistics, 29: 1165) FDR under dependent
multiple hypothesis (motifs abundances and not independent since we hypothesize that groups of
motifs may play a role in the regulation of expression). Filters motifs respecting FDR level
optionally given. Outputs filtered motifs sorted by decreasing FDR-adjusted pvalues on STDOUT.
These tab-separated values on output are:

motif accession (motifID), abundance of the motif in the reference group (abundanceReference),
abundance of the motif in the test group (abundanceTest), FDR-adjusted pvalue of the chi square
test (pvalueFDRAdjusted), trend of abundance for the motif, "over" or "under" ((abundanceTrend),
log base 2 ratio of test group abundance on reference group abundance (abundanceLog2ratio) and
transcripts list targetted by this motif.

Test and reference groups input data should classically originate from the 'analyzeMotifs' program
using '--allDetails' option. This program is part of a toolbox for motifs including detection
(detectMotifs) and analysis (analyseMotifs). Both files must provide each a dataflow with the
following structure (if present, expected column names in parenthesis): a tab-separated file, one line
per motif, with identification (motifID), sequence (motifSeq), sequence length (motifLength), list of
target sequences (targetIDs), list of matching positions in the target sequences (posLists), list of
occurrences per sequences (countsLists), number of sequences targetted (nbTargets), number of
sequences analyzed (allTargets), frequency of sequences targetted by this motif among sequences
(targetFreq), mean match per matching sequence (meanMatch), variance of this mean (varMatch),
median (medianMatch), minimum (minMatch), maximum of this match (maxMatch), total
occurrence on matched sequences (nbMatches), total occurrence of all motifs on all sequences
(allMatches) and proportion of this motif matches over all matches (matchProp) . Any other data on
the line will not be considered (but can be present). If there is no header, these columns are
expected all present and in this order. If header is provided, only a subset of columns can be
provided. However, motifID, targetIDs, nbTargets, allTargets, nbMatches are mandatory.

=head1 OPTIONS

--Help|help|h, produces this help file.
```

--verbose[no-verbose]|Verbose[no-Verbose]|v[no-v], boolean option to print out warnings during execution. Warnings and errors are redirected to STDERR. Defaults to no-verbose (silent mode).

--testGroup|TestGroup|t="filename", mandatory test group filename.

--refGroup|RefGroup|r="filename", mandatory reference group filename.

--fdr|FDR|Fdr|f=0.01, optional value of adjusted pvalues below and equal to which transcripts are considered significant and send to STDOUT. Defaults to 0.01 (1%).

--header|Header|d, boolean option to output a header describing data columns. Defaults to no-header output.

=head1 AUTHORS

Philippe LABEL

=head1 VERSION

1.00

=head1 DATE

24/02/2016

=cut

```
# libraries
use warnings;
use strict;
use Pod::Usage;
use Getopt::Long;
use File::Basename;
use List::Util qw< sum >;
use List::MoreUtils qw< natatime >;
use Statistics::Distributions qw< chisqrprob >;
use Statistics::Multtest qw(:all);
use Bio::PrimarySeq;
use Bio::Tools::IUPAC;
use Data::Dumper;
use 5.10.0;

# scalars
my $help;
my $verbose;
my $debug;                                  # debug purposes only
my $sum = 0;
my $sumRef = 0;
my $testGroup;
my $refGroup;
my $fdrPvaluesOccurrences;
```

```
my $fdrPvaluesTargets;
my $fdrPvaluesFreq;
my $fdrLimit;
my $header;

# lists
my @headersTest;
my @headersRef;

# hashs
my %testGroup;
my %refGroup;
my %chisqPvaluesOccurrences;
my %chisqPvaluesTargets;
my %chisqPvaluesFreq;
my %seqTestGroup;
my %seqRefGroup;

# functions
sub error ($) {
	# management of error messages and help page layout, will stop execution
	# local arguments passed:	1st, error message to output
	my $error = shift;
	my $filename = basename($0);
	pod2usage(-message => "$filename (error): $error Execution halted.", -verbose => 1,
-noperldoc => 1);
	exit(2);
}

sub warning ($) {
	# management of warnings and execution carry on
	# local arguments passed:	1st, warning message to output
	if ($verbose) {
		my $message = shift;
		my $filename = basename($0);
		warn("$filename (info): ".$message."\n");
	}
}

sub debug ($) {
	# management of debugging messages
	# local arguments passed:	1st, warning message to output
	# no return value
	if ($debug) {
		my $message = shift;
		warn(Dumper($message));
	}
}

sub chi_squared_test ($$) {
```

```
	my $observed = shift;
	my $expected = shift;
	my $chi_squared = ($observed - $expected)**2 / $expected;
	my $probability = chisqrprob(1, $chi_squared);
	return $probability;
}

sub find_IUPAC_regex ($) {
	my $sequence = shift;
	my $ambiseq = Bio::PrimarySeq->new(-seq => $sequence, -alphabet => 'dna'); # create the
degenerated dna sequence motif
	my $iupac = Bio::Tools::IUPAC->new(-seq => $ambiseq); # create all alternative motifs
	return $iupac->regexp(); # find the regex corresponding to the degenerated dna sequence
motif
}

MAIN: {
	GetOptions(	"help|Help|h"				=> \$help,
			"verbose|Verbose|v!"			=> \$verbose,
			"header|Header|d!"			=> \$header,
			"testGroup|TestGroup|t=s"		=> \$testGroup,
			"refGroup|RefGroup|r=s"				=> \$refGroup,
			"fdr|FDR|Fdr|f=s"			=> \$fdrLimit,
			"debug!"				=> \$debug
				);
	if ($help) {
		pod2usage(-verbose => 2, -noperldoc => 1);
	exit;
	}

	if ($testGroup) {
		error("Provided filename ('$testGroup') doesn't exist.") if (! -e $testGroup);
		warning("Processing test group from file...");
		open(TESTFILE,$testGroup) or error("Unable to open '$testGroup' file.");
		while (my $line = <TESTFILE>) {
			chomp $line;
			my @items = split("\t",$line);
			if ($items[0] =~ /motifID/) {
				@headersTest = @items;
				next;
			}
			$testGroup{$items[0]} = [@items];
			$seqTestGroup{$items[0]} = $items[1]; # store motif sequence for
comparison with other list
		}
		close TESTFILE or error("Unable to close $testGroup file.");
		warning("...finished");
	}
	else {
		error("Test group filename is mandatory");
	}
	debug(keys(%testGroup));
```

```
	if ($refGroup) {
		error("Provided filename ('$refGroup') doesn't exist.") if (! -e $refGroup);
		warning("Processing reference group from file...");
		open(REFFILE,$refGroup) or error("Unable to open '$refGroup' file.");
		while (my $line = <REFFILE>) {
			chomp $line;
			my @items = split("\t",$line);
			if ($items[0] =~ /motifID/) {
				@headersRef = @items;
				next;
			}
			$refGroup{$items[0]} = [@items];
			$seqRefGroup{$items[0]} = $items[1]; # store motif sequence for
comparison with other list
		}
		close REFFILE or error("Unable to close $refGroup file.");
		warning("...finished");
	}
	else {
		error("Reference group filename is mandatory");
	}
	debug(keys(%refGroup));

	if (join("",@headersTest) ne join("",@headersRef)) {
	error("Both files do not have same columns headers");
	}

	if (! $fdrLimit) {
		$fdrLimit = 0.01;
	}

	warning("Processing differential motifs...");
	my @commonMotifs;
	my @testSpecificMotifs;
	my @refSpecificMotifs;
	foreach my $testItem (keys(%testGroup)) {
		foreach my $refItem (keys(%refGroup)) {
			if($seqTestGroup{$testItem} ~~
(find_IUPAC_regex($seqRefGroup{$refItem}),$seqRefGroup{$refItem})) {
				push(@commonMotifs, ($testItem,$refItem)) ;
			}
			else {
				push(@testSpecificMotifs, $testItem);

			}
		}
	}
	debug(@commonMotifs);
	warning("...finished");
```

```
	warning("Estimating significant abundances for common motifs...");
	my $it = natatime(2,@commonMotifs);
	while (my ($testMotif,$refMotif) = $it->()) {
	debug($testGroup{$testMotif}[8]);
	debug($refGroup{$refMotif}[8]);
	next  if ($testGroup{$testMotif}[8] == 0 or $refGroup{$refMotif}[8] == 0);
		$chisqPvaluesFreq{$testMotif.":::".$refMotif} =
chi_squared_test($testGroup{$testMotif}[8] *100 , $refGroup{$refMotif}[8] * 100);
	}
	$fdrPvaluesFreq = BY(\%chisqPvaluesFreq);  # arguments passed by reference and retrieved
by reference as well, using Benjamini-Yekutieli adjustment
	warning("...finished");

	warning("Outputting structured data...");
	print STDOUT
"motifID\tmotifSeq\tmotifLength\ttestTargetFreq\trefTargetFreq\tfdrAdjPvalue\ttrend\n" if
($header);
	my @sorted = sort ({${$fdrPvaluesFreq}{$a} <=> ${$fdrPvaluesFreq}{$b}} keys(%
$fdrPvaluesFreq));

	foreach my $motif (@sorted) {
		next if (${$fdrPvaluesFreq}{$motif} > $fdrLimit);
		my ($testMotif,$refMotif) = split(":::", $motif);
		print STDOUT "$motif\t";
		print STDOUT $testGroup{$testMotif}[1]."\t";
		print STDOUT $testGroup{$testMotif}[2]."\t";
		print STDOUT $testGroup{$testMotif}[8]."\t";
		print STDOUT $refGroup{$refMotif}[8]."\t";
		print STDOUT sprintf("%.0e",${$fdrPvaluesFreq}{$motif})."\t";
		if ($testGroup{$testMotif}[8] > $refGroup{$refMotif}[8]) {
			print STDOUT "over\t";
		}
		else {
			print STDOUT "under\t";
		}
		print STDOUT "\n";
	}
	warning("...finished");
}	# end of MAIN
```

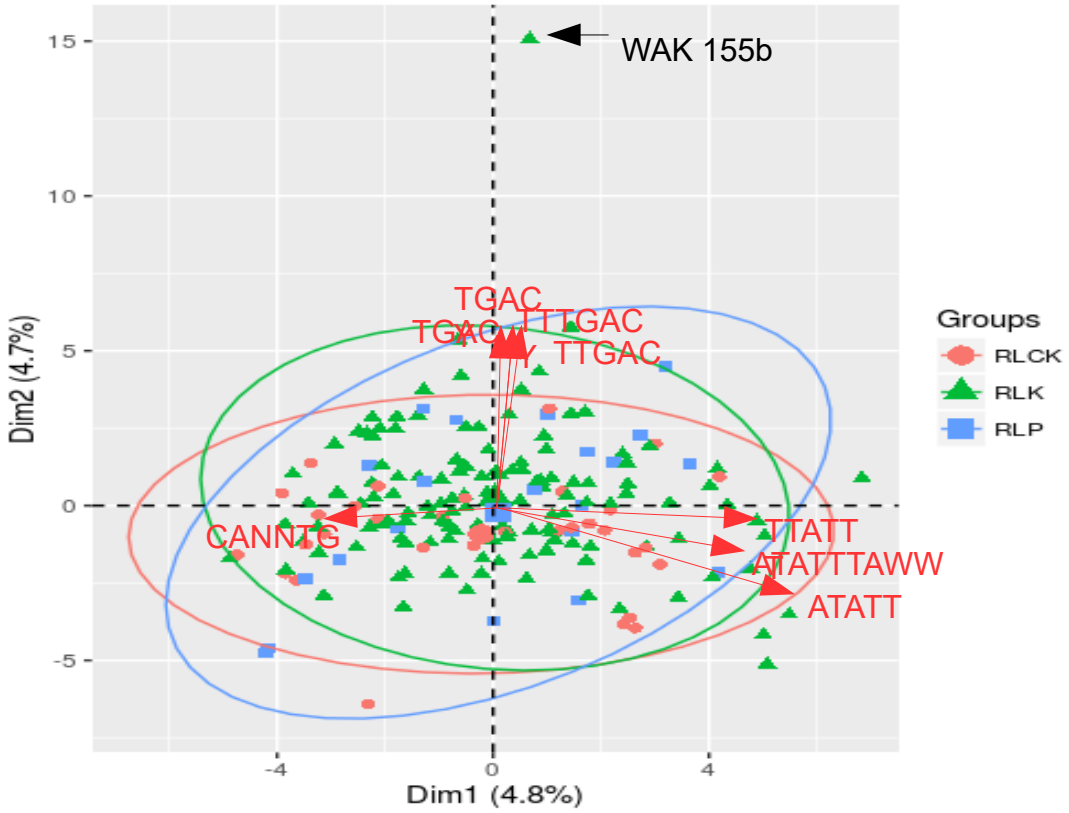

Figure 7': ACP des contributions

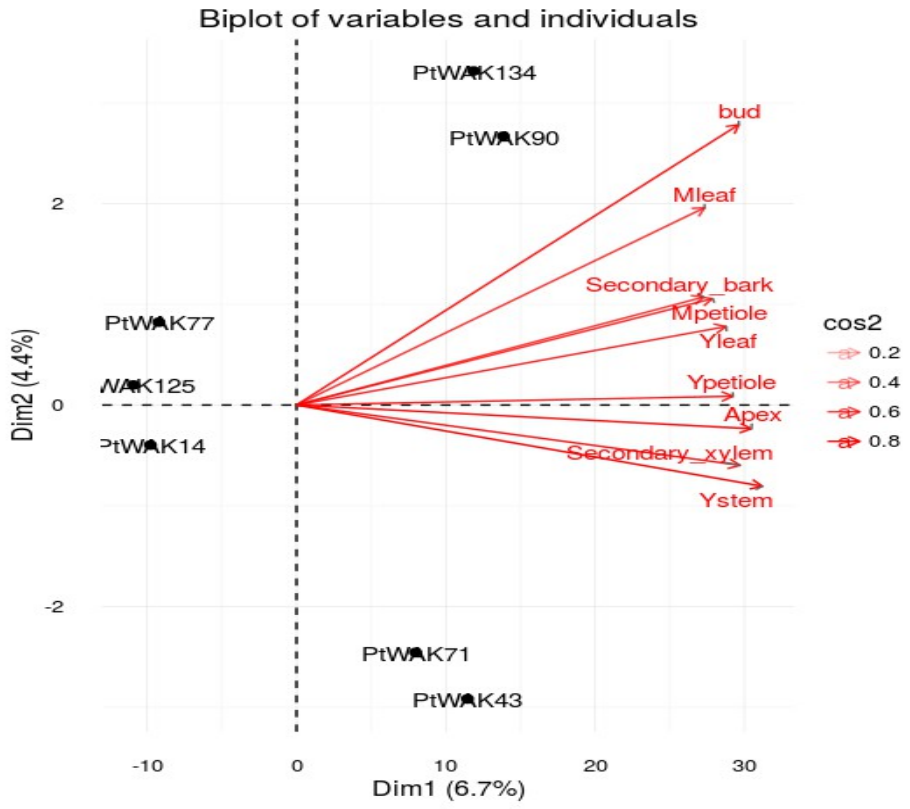

Figure 8': ACP des contributions

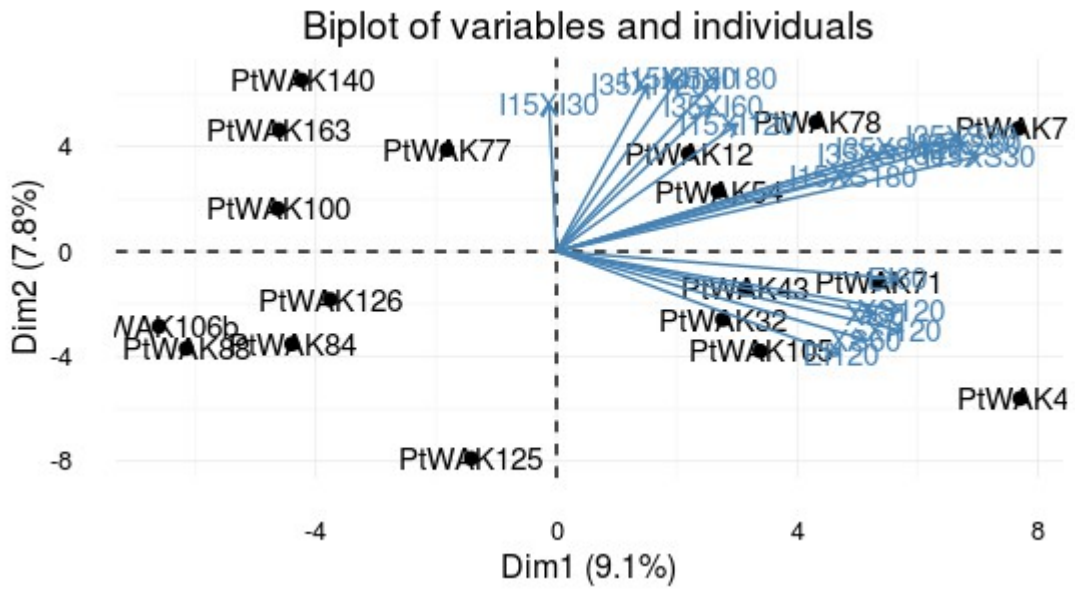

Figure 9': ACP des contributions

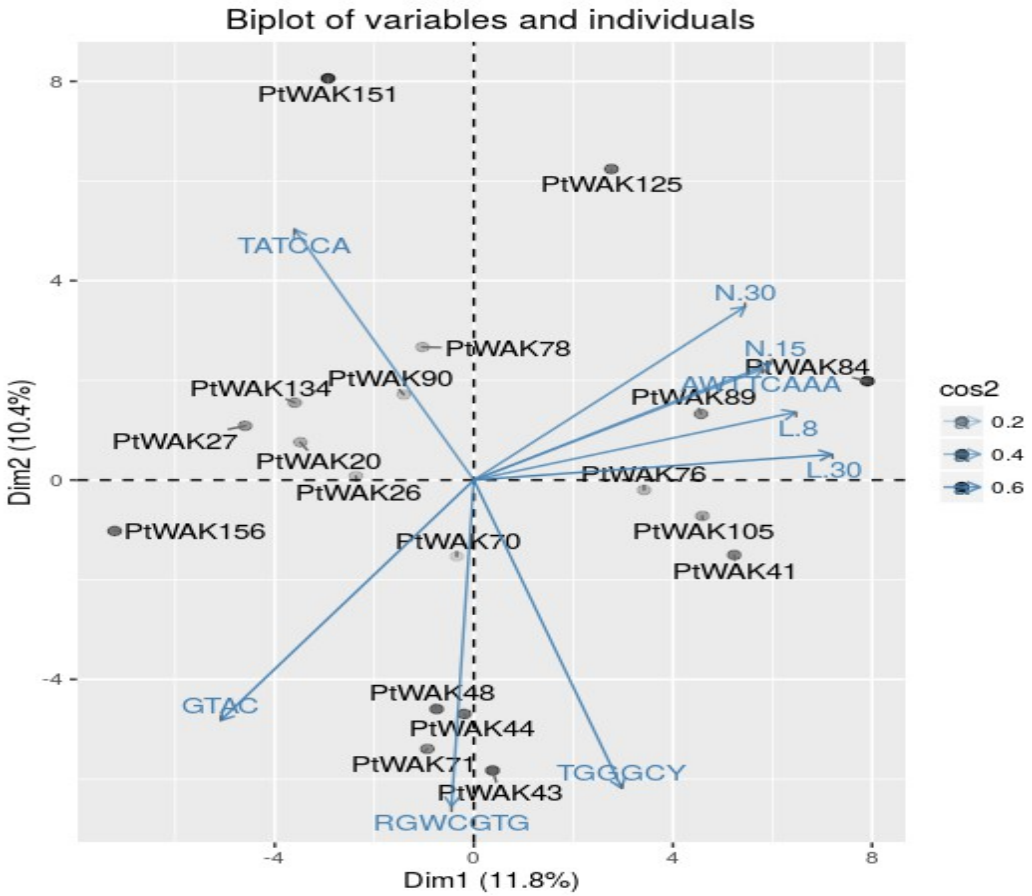

Figure 10': ACP des contributions