Épreuve de mathématiques pratiques et informatique
Rapport du jury – Session 2018

1 Modalités de l'épreuve

Un sujet d'oral est composé d'un unique exercice imposé au candidat, portant sur le programme de mathématiques et d'informatique de BCPST. Dans la mesure du possible, les sujets ont été choisis pour couvrir l'ensemble du programme des deux années (BCPST1 et BCPST2).

Les candidats disposent de 30 minutes de préparation dans une salle dédiée, où ils ont accès à un ordinateur sur lequel sont installés les logiciels Pyzo et Spyder (Python 3), Excel et Geogebra.

Les candidats disposent d'une clé USB fournie par le concours au début de leur préparation, sur laquelle il leur est demandé d'enregistrer leur travail.

À l'issue de la préparation, ils sont accompagnés dans une salle d'interrogation (5 minutes au maximum sont prévues pour la transition), où ils exposent pendant environ 18-20 minutes le résultat de leur travail de préparation et ils dialoguent avec l'examinateur. Dans cette salle de passage, ils disposent également d'un ordinateur connecté à un vidéoprojecteur, sur lequel ils peuvent exécuter les programmes sauvegardés sur la clé USB fournie.

À l'issue de ce premier temps d'interrogation, les candidats sont accompagnés vers une deuxième salle, dans la mesure du possible voisine de la précédente, où ils présentent le résultat de leur projet informatique durant 18-20 minutes devant un second examinateur. Dans cette deuxième salle, les candidats disposent d'un ordinateur avec vidéoprojecteur sur lequel est affiché le dossier qu'ils ont au préalable envoyé au concours au début du mois de juin.

2 Éléments statistiques

Un résumé statistique est fourni ci-dessous, pour chacune des parties de l'épreuve ainsi que pour la note finale (chaque partie était notée sur 10). On notera que la corrélation entre la note d'un candidat sur la partie Mathématiques pratiques et sa note sur la partie Projet informatique reste comme les années précédentes assez faible, même si en légère augmentation ($r = 0{,}43$, $r^2 = 0{,}19$). En particulier, d'assez nombreux candidats ayant de grosses difficultés en mathématiques ont pu montrer des compétences très intéressantes en informatique.

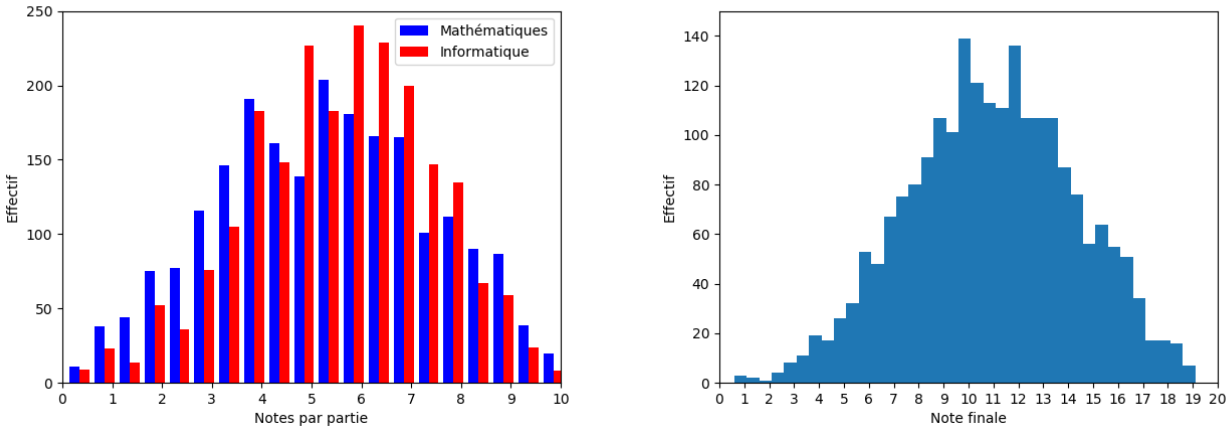
Les distributions des notes sont semblables aux années précédentes concernant chacune des parties, mais les notes globales obtenues sont un peu moins dispersées, les notes extrêmes sur 20 ayant été faibles (très peu de candidats ont eu une note maximale (ou minimale) dans les deux épreuves).

Partie	Moyenne	Médiane	Écart-type	$0 \leqslant x < 2$	$2 \leqslant x < 4$	$4 \leqslant x < 6$	$6 \leqslant x < 8$	$8 \leqslant x \leqslant 10$
Mathématiques	5,41	5,5	2,14	8%	24%	32%	25%	11%
Informatique	5,71	6	1,84	4%	18%	37%	33%	8%

Résumé statistique par partie de l'épreuve

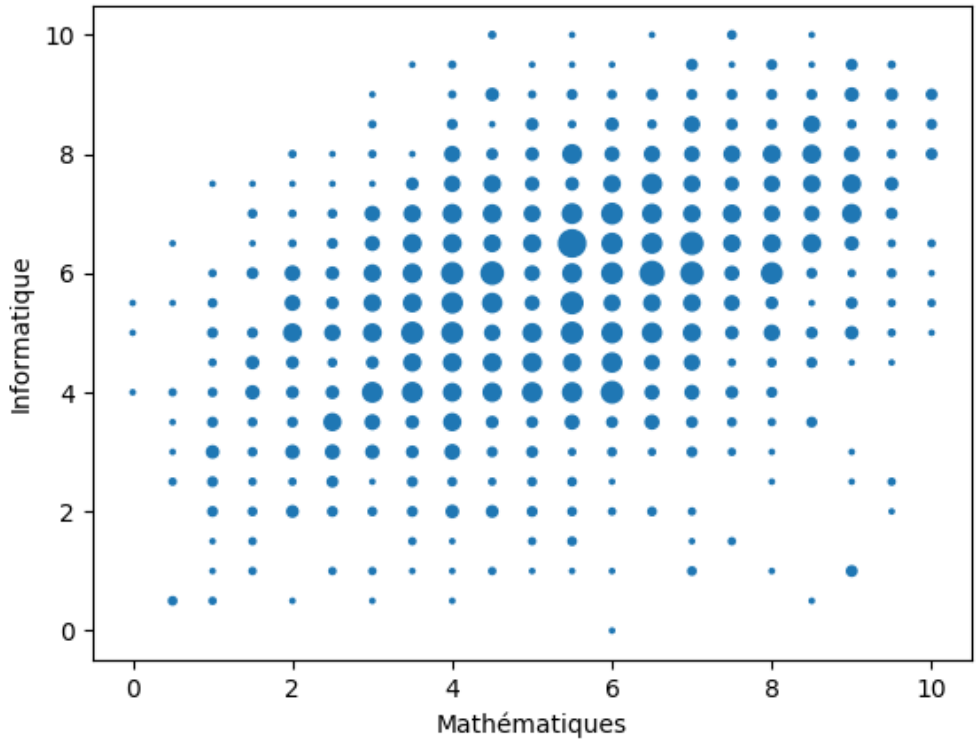
	Moyenne	Médiane	Écart-type	$0 \leqslant x < 4$	$4 \leqslant x < 8$	$8 \leqslant x < 12$	$12 \leqslant x < 16$	$16 \leqslant x \leqslant 20$
Note finale	11,12	11	3,38	2%	18%	42%	31%	7%

Résumé statistique général



Histogrammes des notes

Dans le graphique suivant, la taille du disque de centre (x, y) est proportionnelle au nombre de candidats ayant obtenu la note x à la partie Mathématiques pratiques et y à la partie Informatique. Comme dit plus haut, les notes sont assez faiblement corrélées ; ce graphique est encore très similaire à celui des trois années précédentes.



3 Partie Mathématiques Pratiques

3.1 Ce qui est attendu des candidats en Mathématiques Pratiques

Dans un but d'équité, tous les candidats convoqués à une même heure préparent le même sujet, quel que soit leur jury d'interrogation. Les examinateurs connaissent les énoncés sur lesquels portent l'interrogation, il est donc inutile **et même déconseillé** de relire l'énoncé.

Pendant la période d'interrogation, un dialogue entre le candidat et l'examinateur s'instaure. L'oral est un échange, n'est pas un écrit, et en aucun cas n'est une conférence du candidat face à un jury muet. Les interrogateurs posent des questions visant à évaluer les compétences du candidat, et cherchent avec bienveillance à l'aider pour compléter son argumentation ou se rendre compte d'une erreur. Elles ne constituent en aucun cas des pièges tendus. Le jury attend de la bonne volonté et une attitude ouverte pour répondre aux questions.

Il n'est pas indispensable d'avoir traité la totalité de l'exercice pour obtenir une excellente note. Il est préférable d'avoir mené un raisonnement rigoureux et argumenté, reposant sur des connaissances solides, plutôt que d'avoir donné tous les résultats (même justes), trop vite et sans explication réelle.

Depuis cette année, chaque sujet comporte une question de cours, qui précède l'énoncé de l'exercice à préparer. Cette question est indépendante de l'exercice, dans le sens où elle peut concerner un thème différent de celui abordé dans la suite.

Au début de l'interrogation (en général, juste après la question de cours), le candidat peut être invité par l'examinateur à préciser la liste des questions qu'il a eu le temps d'aborder pendant la préparation, mais nous conseillons aux candidats de le faire spontanément. Ceci n'a pas pour but d'évaluer *a priori* le candidat, mais permet simplement à l'examinateur de gérer au mieux le déroulement de l'oral, et d'éviter que certaines questions préparées par le candidat n'aient pas le temps d'être exposées. Pour le jury, « aborder une question » pendant la préparation signifie que le candidat en a traité au moins une partie, et ne s'est pas contenté d'essayer et achopper. Certains candidats maladroits affirment avoir « abordé » beaucoup de questions, alors qu'ils ne les ont en fait que survolées, ce qui est ensuite pénalisant pour la gestion du temps de l'oral.

Chaque sujet comporte au moins une question d'informatique et les examinateurs interrogent systématiquement sur au moins une de ces questions, qu'elle ait été préparée ou non par le candidat en amont. Le cas échéant, le jury peut poser une question élémentaire pour vérifier les compétences du candidat en informatique.

La durée de l'interrogation orale doit impérativement être respectée. Lorsque l'examinateur annonce la fin du temps réglementaire, il est attendu que le candidat efface efficacement le tableau et range rapidement ses affaires, afin de ne pas pénaliser le candidat suivant.

3.2 La question de cours

Comme nous l'avions annoncé dans le rapport 2017, la nouveauté cette année consistait en l'introduction d'une question de cours en début de chaque sujet. La question de cours est écrite sur le sujet (et non posée à l'oral par l'examinateur), le candidat en prend donc connaissance au début de sa préparation avec le reste du sujet.

Les examinateurs ont constaté à plusieurs reprises des candidats n'ayant pas préparé cette question et semblant la découvrir lorsque l'examinateur leur demande de l'exposer. Il serait bon, de manière générale, que les candidats lisent bien attentivement le sujet lors de leur préparation, et se préparent de manière adéquate durant l'année à cette question de cours.

Les seules questions de cours qui peuvent être posées sont des énoncés de définitions ou de théorèmes, à l'exclusion de toute démonstration. La rédaction de ces questions s'appuie en priorité sur le Programme Officiel de BCPST (rentrée 2013) et peut couvrir l'ensemble du programme des deux années de classe préparatoire. Le but est simplement de vérifier brièvement la connaissance d'un point du cours, et ne doit

pas donner lieu à un développement trop long. Si cela est pertinent, le candidat peut se contenter d'une réponse orale. Cependant, le candidat doit être précis lorsqu'il énonce une propriété. Si un théorème est demandé, les hypothèses requises doivent être rappelées par le candidat.

Dans la plupart des cas, l'examinateur se contente d'écouter la question de cours, sans la reprendre avec le candidat même si elle est partiellement ou totalement incorrecte. Cette question a pour première ambition de pouvoir valoriser un candidat sérieux mais ayant des difficultés sur le reste du sujet. Elle a aussi pour but d'inciter les candidats à travailler régulièrement l'apprentissage de leur cours, ceci sur leurs deux années de classe préparatoire, ce qui donne bien entendu un poids supplémentaire aux consignes des professeurs de BCPST données à leurs élèves sur l'importance d'avoir une bonne connaissance du cours. Nous sommes tous convaincus qu'une meilleure maîtrise du cours aura une répercussion immédiate chez les candidats sur leurs capacités à résoudre les exercices proposés à l'oral.

Les questions sont choisies de manière aléatoire, avec remise éventuelle lors d'une même session. Il est donc inutile pour certains candidats de faire l'impasse sur une question déjà tombée lors d'une session passée ou en cours, cette question pouvant être réutilisée de manière immédiate ou future. Nous publions ci-après une liste **non exhaustive** d'exemples de questions de cours utilisées lors de la session 2018, par souci d'équité entre les différentes classes préparatoires. Cette liste donne simplement un aperçu des différentes attentes du jury, mais certaines de ces questions pourront (ou non) être reprises dans les sessions futures, comme complétées par de nouvelles.

Comme pour les exercices, le jury ayant bien conscience que les questions de cours sont de difficulté variable, il en tient compte dans son évaluation. La part de la note finale accordée à cette question peut donc être ajustée d'un sujet à l'autre, l'évaluation de la connaissance du cours restant globale sur l'ensemble de l'interrogation.

Questions de cours 2018 très bien traitées
Définition du module d'un nombre complexe.
Allure de la représentation graphique de la fonction Arctan en précisant ses asymptotes en $-\infty$ et $+\infty$.
Inversibilité d'une matrice carrée 2×2.
Soit $(a, b) \in \mathbb{R}^2$ tel que $a^2-4b > 0$. Donner l'ensemble des solutions de l'équation différentielle : $y''+ay'+by = 0$.
Allure des représentations graphiques des fonctions exponentielle et logarithme népérien.
Énoncer le théorème d'intégration par parties sur une intégrale.
Donner la définition de la convergence d'une intégrale généralisée $\int_a^{+\infty} f(t)dt$ si f est une fonction continue sur l'intervalle $[a, +\infty[$.
Quelles sont les solutions de l'équation différentielle $y' + a(t)y = 0$?
Énoncer l'inégalité de Bienaymé-Tchebychev.
Donner la définition d'une base d'un espace vectoriel.
Pour $\|q\| < 1$, donner l'expression des sommes suivantes : $\sum_{n=0}^{+\infty} q^n$, $\sum_{n=1}^{+\infty} nq^{n-1}$ et $\sum_{n=2}^{+\infty} n(n-1)q^{n-2}$
Définition du noyau d'une application linéaire $f : E \to F$.
Définition d'une famille libre $(u_1, u_2, \ldots, u_k)$ de vecteurs dans un espace vectoriel E.
Quand dit-on qu'une variable aléatoire suit la loi géométrique de paramètre p ? Quelles sont alors son espérance et sa variance ?

Questions de cours 2018 traitées de façon plus approximative
Définition de l'espérance d'une variable aléatoire X discrète à valeurs dans $\mathbb{N}$.
Rappeler la valeur de $P(X = k)$ si X suit une loi hypergéométrique.
Définition d'une base orthonormale de $\mathbb{R}^n$.
Donner la définition d'une valeur propre et d'un vecteur propre pour un endomorphisme f d'un espace vectoriel E de dimension finie.
Définition d'une matrice carrée inversible.
Densité d'une loi normale centrée réduite.
Définition de l'espérance d'une variable aléatoire X admettant une densité f continue sur $\mathbb{R}$.
Donner le développement limité au voisinage de 0, à l'ordre 4 de la fonction $x \mapsto \dfrac{1}{1+x}$.
Comment déterminer les lois marginales d'un couple (X, Y) de variables aléatoires réelles discrètes si on connaît la loi conjointe ?
Définition d'une application $f : E \to F$ injective.
Définition de la variance d'une variable aléatoire discrète admettant un moment d'ordre 2.
Donner la définition du produit scalaire de deux vecteurs $(x_1, \ldots, x_n)$ et $(y_1, \ldots, y_n)$ de $\mathbb{R}^n$.
Allure de la représentation graphique d'une densité de la loi exponentielle de paramètre 1.
Énoncer le théorème de Rolle.
Donner la définition d'un endomorphisme d'un espace vectoriel E.
Définition d'une application $f : E \to F$ surjective.
Énoncer l'inégalité de Markov.
Développement limité de $\sin(x)$ à l'ordre 5 au voisinage de 0.
Lien(s) entre l'indépendance de deux variables aléatoires discrètes et leur covariance.
Définition de la dérivée d'une fonction f en un point a.
Condition nécessaire et suffisante pour qu'une application linéaire soit injective.
Rappeler la formule des accroissements finis.
Allure de la représentation graphique d'une densité de la loi normale d'espérance 1 et de variance 1.
Définition de la covariance de deux variables aléatoires réelles discrètes.
Somme et produit des racines d'une équation du second degré $ax^2 + bx + c = 0$ avec $(a, b, c) \in \mathbb{R}^3$, $a \neq 0$.
À quelle(s) condition(s) sur sa fonction de répartition une variable aléatoire X admet-elle une densité de probabilité ? Comment détermine-t-on alors une densité de X ?
Qu'appelle-t-on *racine* d'un polynôme ? Qu'appelle-t-on *ordre de multiplicité* d'une racine d'un polynôme ?
Énoncer une condition suffisante et non nécessaire pour qu'une matrice soit diagonalisable, puis une condition nécessaire et suffisante pour qu'une matrice soit diagonalisable.
Soit X une variable suivant la loi de Poisson de paramètre $\lambda > 0$. Expliciter la loi de X, et donner son espérance et sa variance.

Questions de cours 2018 ayant posé problème aux candidats
Définition d'une famille génératrice dans un espace vectoriel E.
Densité d'une loi exponentielle de paramètre λ.
Définition et convergence de la somme de Riemann d'une fonction f continue sur $[0,1]$.
Soit u, un endomorphisme de $\mathbb{R}^n$. Donner une condition nécessaire et suffisante pour que u soit diagonalisable.
Énoncer la formule des probabilités totales.
Soient $\mathcal{B}_1$ et $\mathcal{B}_2$ deux bases d'un espace vectoriel E de dimension n. On appelle P la matrice de passage de $\mathcal{B}_1$ à $\mathcal{B}_2$. Si $\overrightarrow{x}$ est un vecteur de E, quelle relation lie $mat_{\mathcal{B}_1}(\overrightarrow{x})$ et $mat_{\mathcal{B}_2}(\overrightarrow{x})$?
Définition de la négligeabilité d'une fonction par rapport à une autre au voisinage de $+\infty$.
Énoncer le théorème du rang pour une application linéaire $f : E \to F$.
Énoncer la définition de l'indépendance de deux variables aléatoires réelles discrètes.
Donner la définition d'une valeur propre ainsi que d'un sous-espace propre pour une matrice $A \in \mathcal{M}_n(\mathbb{R})$.
Énoncer la définition et le théorème des suites adjacentes.
Croissances comparées entre les suites puissance n^α (avec $\alpha > 0$), géométrique a^n (avec $a > 1$) et factorielle $n!$.
Si α est un réel fixé, rappeler les solutions de l'équation $\cos(x) = \cos(\alpha)$, d'inconnue $x \in \mathbb{R}$.
Définition de la notion d'indépendance mutuelle d'une famille finie d'événements.
Énoncer le théorème central limite.
Donner la définition de la partie entière d'un réel.
Équation cartésienne d'un plan de $\mathbb{R}^3$ normal au vecteur $u = (1, 2, -1)$.
Énoncer la formule des probabilités composées.
Énoncer le théorème de transfert dans le cas d'une variable aléatoire admettant une densité.
Définition d'un point critique pour une fonction de deux variables.
Donner une représentation paramétrique de la droite de l'espace passant par le point A de coordonnées (x_A, y_A, z_A) et dirigée par le vecteur $\overrightarrow{u}$ de coordonnées (a, b, c).
Énoncer le théorème de transfert dans le cadre d'une variable aléatoire réelle discrète.
Énoncer la loi faible des grands nombres.
Développement limité à l'ordre 4 de $\ln(1 + x)$ lorque x est au voisinage de 0.
Donner la formule de Taylor-Young.
Énoncer la formule de Bayes.
Dérivation d'une fonction de la forme $f(x(t), y(t))$, la fonction f étant de classe $\mathcal{C}^1$ et les fonctions x et y étant dérivables.
Énoncer l'inégalité de Cauchy-Schwarz.
Énoncer le théorème de Pythagore dans $\mathbb{R}^n$.

3.3 Remarques générales sur la forme de l'oral Mathématiques Pratiques

- Pour cette quatrième année où cet oral est mis en place, nous avons encore une fois fortement apprécié que les candidats soient bien préparés à l'épreuve. Nous tenons à féliciter les préparateurs qui ont manifestement tenu compte des remarques des rapports précédents dans la préparation de leurs étudiants.

- Le niveau général des candidats est plutôt bon, notamment en probabilités. Peu de candidats sont en grande difficulté face aux notions du programme. La connaissance du cours s'est globalement améliorée, notamment grâce à l'introduction de la question de cours, sans être toutefois totalement satisfaisante.

- L'énoncé des définitions et des théorèmes au programme doit être précis, notamment sur le rappel des bonnes hypothèses. Le jury se réserve le droit de poser une question de cours à tout moment de l'interrogation s'il le juge nécessaire, pour s'assurer de la bonne compréhension du candidat. L'évaluation du cours n'est pas limitée à la question posée en début de sujet.

- Le jury rappelle aux candidats que les calculs qui ont pu être réalisés lors de la préparation ne doivent pas forcément être totalement développés au tableau pendant l'interrogation. Les candidats peuvent avoir une part de recul et de synthèse vis-à-vis de leurs brouillons, afin de gagner du temps lors de leur passage dont la durée est limitée.

- Le jury tient à encourager les candidats à émettre un avis critique sur leurs résultats lorsqu'ils sont clairement incohérents (tableau de variations en désaccord avec les limites, obtention d'une probabilité supérieure à 1, etc.). S'aider d'un graphique ou d'un schéma est fortement valorisé dans certaines situations par les examinateurs. On peut déplorer qu'en particulier les tracés de courbes qui peuvent être demandés spontanément à l'oral, notamment sur les fonctions usuelles, sont trop souvent erronés, alors qu'ils peuvent permettre à des candidats de visualiser des erreurs manifestes dans leurs raisonnements ou leurs conclusions. Le jury demandera par conséquent aux candidats plus souvent dans les années à venir de faire des tracés de représentations graphiques, par exemple dans les questions de cours.

- Faire l'impasse sur une ou plusieurs parties du programme de mathématiques ou d'informatique peut avoir de lourdes conséquences sur la prestation du candidat et peut être très pénalisant pour son évaluation. C'est donc bien entendu une stratégie à prohiber.

- Les candidats doivent enfin pouvoir s'adapter à des sujets d'oraux parfois originaux, de longueur très diverse, notamment ceux où il est attendu un peu de modélisation. Les examinateurs connaissent les sujets et leur difficulté relative, et adaptent leurs attentes et la notation en conséquence. Certains candidats sortent de leur préparation en ayant traité très peu de questions, et commencent leur oral en étant déstabilisés. L'examinateur trouvera toujours le moyen d'évaluer lors de la discussion les connaissances du candidat en le guidant dans la résolution ou en lui posant des questions supplémentaires.

  En particulier, lorsque le sujet comporte une modélisation ou plusieurs définitions, et que le sujet comporte un en-tête un peu long, nous encourageons les candidats à consacrer du temps à essayer d'assimiler au mieux les notations introduites afin de bien comprendre le sujet, et ne pas se ruer sur la résolution des questions au risque de passer à côté d'informations importantes, voire d'avoir mal compris le cadre de l'exercice. Il vaut mieux aborder moins de questions pendant la

préparation, mais s'être bien imprégné des notions mises en jeu pour bien suivre les indications de l'examinateur lors de l'interrogation.

3.4 Remarques générales sur le contenu

Algèbre. En 2018 et comme pour les années précédentes, la plupart des exercices proposés en algèbre linéaire et bilinéaire étaient élémentaires et leur résolution a été trop souvent décevante pour un trop grand nombre de candidats. La cause de ces échecs était souvent due à une connaissance trop approximative du cours et à un manque de réflexes méthodologiques face aux questions posées par les sujets.
Par exemple, pour de nombreux candidats, la seule méthode envisageable pour parler d'une valeur propre λ est d'étudier le rang de $A - \lambda I_n$, même lorsque cette valeur propre est donnée dans l'exercice, ou quand l'inversibilité (ou non) de la matrice a été établie précédemment.
Il est donc dommage que les candidats perçoivent les notions d'algèbre comme cloisonnées sans tenir compte du contexte de l'exercice, alors que l'intitulé des questions peut leur suggérer des méthodes plus appropriées et leur évitent des calculs fastidieux. Nous espérons que les futurs candidats sauront tenir compte de nos remarques pour les sessions à venir.

Probabilités. Les candidats sont plutôt à l'aise avec les exercices de probabilités discrètes, tout en présentant des difficultés dès que l'on sort du cadre fini. Les candidats semblent se référer, voire se limiter à leurs connaissances acquises en première année, et mal percevoir l'évolution étudiée dans le programme de deuxième année sur les variables aléatoires discrètes infinies.
Nous avons apprécié cette année que les exercices comportant des produits de convolution (dont la formule est systématiquement rappelée dans les sujets) soient mieux maîtrisés par les candidats, ce qui montre une nouvelle fois que les préparations tiennent bien compte de nos remarques.
Cependant, les questions classiques du type « déterminer la loi de $f(X)$ » lorsque X est une variable à densité et f une fonction simple, ont posé des problèmes insurmontables à un trop grand nombre de candidats. Très peu s'intéressent au support de $f(X)$ avant de travailler sur la fonction de répartition, manquent de rigueur sur la manipulation des événements et sur les éventuels cas à traiter.
Parfois les candidats lisent à moitié les questions et répondent de ce fait à une question qui ne leur est pas posée. Par exemple, les questions « montrer que f est une densité de la variable X » (avec X précédemment définie) ou « montrer que f est une densité de probabilité » ne se traitent pas de la même manière.

Analyse. Cette année, nous avons remarqué que lors d'une question du type « montrer la convergence de la suite (u_n) », les candidats ne pensent pas spontanément à la méthode classique de la « convergence monotone ». Nous avons apprécié cependant que les candidats soient plus à l'aise avec les sommes de Riemann, et les calculs de dérivées et primitives.
Nous pouvons regretter toutefois que sur les exercices plus fins d'analyse, on note de nombreuses lacunes sur la manipulation des équivalents et sur les formules des développements limités usuels.

Informatique. Les candidats montrent chaque année une nette amélioration, témoin d'un excellent accompagnement par leurs enseignants pour acquérir les compétences du langage Python.
Cependant, nous avons vu resurgir des erreurs aperçues lors de la première session de 2015, qui avaient été corrigées depuis chez la plupart des candidats. En particulier, l'utilisation de `range` était parfois approximative, ou les confusions entre la simulation d'une variable aléatoire à densité et le calcul de sa fonction de répartition (ou de sa densité) étaient réapparues.
Globalement, les tracés de courbes en Python sont encore une fois très peu corrects. C'est pourtant

un exercice classique qui nous semble être un objectif à atteindre par chaque étudiant ayant travaillé en Python. Nous ne pouvons que conseiller aux candidats de niveau modeste et ayant des difficultés à maîtriser Python, de savoir le cas échéant a minima utiliser le logiciel Geogebra. En effet, ce dernier, sans avoir besoin d'être expert dans son usage, permet également de tracer rapidement des courbes à moindre coût.
Signalons une nouvelle fois que les sujets conseillent parfois d'utiliser des bibliothèques. Ce ne sont que des indications, les candidats peuvent très bien utiliser une autre méthode que celle proposée, et peuvent montrer toute l'autonomie qu'ils ont pu acquérir en deux (ou trois) ans d'informatique.
Il serait bon que les candidats se préparent pendant l'année à savoir « présenter » rapidement un programme qu'ils ont réalisé, le temps d'interrogation étant très rapide au concours. Il est par exemple inutile de lire à l'oral la liste des bibliothèques qu'ils ont importé, ou de relire le programme ligne par ligne à l'examinateur. Il vaut mieux être concis et expliquer le fonctionnement du programme de manière synthétique.
Lorsqu'un programme utilise des simulations aléatoires, il est naturel que les candidats exécutent leur programme devant l'examinateur plusieurs fois, rien que pour témoigner que les résultats peuvent varier d'une exécution à l'autre. À cette fin, nous encourageons les candidats à se familiariser de façon optimale à l'usage de Pyzo ou Spyder, en particulier la gestion de la console et les commandes rapides d'exécution de leurs programmes, afin de gagner du temps lors de l'oral.
Enfin, certains candidats arrivent devant l'examinateur avec une clé USB vide. Il vaut toujours mieux pour un candidat avoir un début de programme sur sa clé USB à présenter, quitte à expliquer ensuite à l'examinateur comment il aurait achevé son programme à l'oral. Les candidats doivent en tout cas avoir en tête qu'il est toujours possible de modifier ou corriger leurs programmes pendant l'interrogation elle-même, l'ordinateur restant à leur entière disposition durant tout leur passage.
Par ailleurs, pour les candidats ayant réalisé un programme complet, il est dommage que certains d'entre eux n'essaient pas de l'exécuter au moins une fois lors de leur préparation. Cela leur permettrait peut-être de prendre conscience d'une erreur de syntaxe manifeste, facile à corriger.

3.5 Conclusion

Le niveau des candidats est très hétérogène, avec tous les intermédiaires entre le candidat brillant et le candidat n'ayant quasiment aucune connaissance ni compétence. L'examinateur s'attache à poser des questions adaptées pour évaluer le niveau du candidat de manière circonstanciée.
Le ressenti de certains candidats n'est pas toujours en adéquation avec la note obtenue. Tous les jurys s'efforcent d'être aimables et bienveillants, mais n'en ont pas moins pour mission d'évaluer et de classer les candidats. Les examinateurs sont tenus à un devoir de réserve quant à l'évaluation du candidat, ce qui peut parfois être interprété à tort par certains candidats pour de la froideur ou de l'indifférence. Un candidat peut ressortir déçu de son interrogation et peut néanmoins obtenir une note tout à fait satisfaisante, et inversement.
L'ensemble des examinateurs a une fois de plus apprécié l'attitude respectueuse et courtoise de tous les candidats, ce qui a facilité le bon déroulement des oraux de cette session 2018.

4 Partie Informatique

4.1 Modalités de l'épreuve

La partie de l'épreuve consacrée au projet informatique dure vingt minutes. Les candidats disposaient de sept minutes, au maximum, pour présenter leur projet, suivies d'un entretien. Le jury est très satisfait de ce format, qui permet de bien juger de la maîtrise du projet par le candidat, mais également d'évaluer de manière plus générale ses compétences en informatique, conformément au programme d'informatique des classes de BCPST. Le jury constate que les candidats sont, pour une très large majorité, très bien préparés au format de l'épreuve.

Pour la partie présentation, dans le cadre explicité par la notice du concours, il appartient au candidat d'insister sur ce qui fait l'intérêt scientifique et algorithmique de son projet :

- intérêt des structures de données utilisées, en regard de la situation étudiée (modélisation) ou des algorithmes utilisés,
- solution algorithmique élégante,
- qualité des résultats obtenus,
- etc.

Pendant l'entretien, les examinateurs cherchent à évaluer la maîtrise du candidat et son implication dans son projet. Cela peut se faire, suivant les cas, en posant des questions d'ordre général sur le contexte algorithmique ou scientifique du projet, ou des questions précises sur le code python présenté par le candidat :

- Quels sont les arguments de cette fonction ?
- Que renvoie-t-elle si on l'appelle sur telle valeur ?
- Comment pourrait-on ajouter telle fonctionnalité ?
- Justifier et/ou décrire les structures de données.
- Où dans le code est implémentée telle fonctionnalité ?
- etc.

La maîtrise du programme d'informatique enseigné en BCPST est le plus souvent évaluée en demandant aux candidats de traiter au tableau un exercice de programmation. Le plus souvent, cet exercice était basé sur le projet :

- écrire une fonction parcourant une structure de données du projet,
- écrire une fonction à l'aide des fonctions du projet,
- reprogrammer dans un cas plus simple une fonction du projet,
- rendre une fonction plus générale ou plus efficace,
- ré-écrire de manière plus concise une partie du projet traitée maladroitement,
- etc.

mais il pouvait aussi être sans rapport direct si celui-ci ne se prêtait pas à ce type de question :

- programmer récursivement de la factorielle (si la récursivité a été utilisée dans le projet),
- écrire la définition d'une classe très simple (si celle-ci a été utilisée),
- écrire une fonction testant une propriété des matrices ou des chaînes de caractère (si de tels objets ont été utilisés dans le projet),
- programmer une fonction relative strictement au programme de BCPST (le plus souvent exercice basique sur les listes).

Les candidats étaient, dans l'ensemble, bien préparés à ce type d'exercice.

Les candidats présentant un projet ambitieux et apparemment bien maîtrisé, mais incapables d'écrire au tableau une fonction élémentaire (déterminer le maximum d'une liste, compter le nombre d'occurrences d'une valeur dans une structure de données par exemple) ont été fortement pénalisés.

4.2 Remarques générales

Nous reprenons ici une grande partie des points évoqués dans les rapports précédents. Les attentes du jury sont maintenant globalement comprises et peu de projets très faibles ont été présentés cette année.

- La très grande majorité des candidats semble s'être investie dans leur projet, et avoir, ce faisant, acquis des compétences qui leur seront utiles tant dans la suite de leurs études que dans leur vie professionnelle.
- La maîtrise générale des projets était globalement bonne.
- Les candidats veilleront à garder un esprit critique quant à la portée des programmes qu'ils ont réalisés. En particulier, il est de bon aloi de se demander si les objectifs que l'on s'était fixés initialement sont remplis ou non.
- Les présentations orales, malgré quelques défauts récurrents évoqués plus bas, étaient globalement satisfaisantes et illustrées de schémas clairs et bien choisis.
- Aucun problème majeur d'organisation n'a été constaté cette année.

4.3 Remarques sur la présentation

Lors de la phase de présentation du projet, le candidat dispose du diaporama au format PDF qu'il a déposé préalablement par voie électronique et qui sert de support à sa présentation. Le dossier déposé par le candidat est directement mis à disposition sur l'ordinateur utilisé par le candidat, mais il est rappelé que les candidats doivent venir avec une clé USB de secours contenant leurs fichiers en cas de problème informatique. L'usage de notes est proscrit. Plusieurs candidats n'ont pas déposé correctement leurs fichiers par voie électronique (fichier powerpoint au lieu de pdf, fichier python incomplet, voire absence de fichiers) : ces candidats ont été pénalisés, même s'ils ont pu en général utiliser leur clé de secours.

Une bonne présentation doit être synthétique. Nous attirons l'attention des candidats sur le fait que la présentation des structures de données et algorithmes utilisés est cruciale. Le jury apprécie que les candidats présentent l'enjeu de leur projet, les hypothèses qui ont été faites dans la modélisation choisie (le cas échéant), l'architecture générale du projet et une analyse des résultats et des perspectives ultérieures.

Le jury a apprécié que les candidats n'aient pas passé trop de temps à présenter des concepts non-informatiques (biologiques, physiques, mathématiques, etc.). Les candidats ont généralement pris soin d'expliciter leurs modèles et/ou les règles des jeux étudiés. Nous rappelons aux candidats que l'examinateur n'est pas censé interrompre le candidat durant cette phase ; il faut donc éviter de se retrouver dans une situation où l'examinateur ne voit pas où le candidat veut en venir. Pour les mêmes raisons, les présentations contenant de trop nombreuses diapositives ou présentées en récitant un texte à toute vitesse pour tenir dans les sept minutes imposées sont à proscrire. Pour un jeu, demander à l'examinateur s'il connaît déjà les règles ou s'il souhaite qu'on les lui explique rapidement semble une idée raisonnable ... à condition de tenir compte de sa réponse.

Il est fortement recommandé de commenter une ou deux fonctions présentant un intérêt algorithmique durant cette phase de présentation. Il est donc souhaitable que ces fonctions :

- fassent apparaître des algorithmes non triviaux,
- soient concises ou bien découpées en blocs dans le diaporama, afin qu'elles soient facilement lisibles.

Le jury a apprécié que, dans leurs diapositives, de nombreux candidats utilisent des flèches, des encadrements, des accolades ou d'autres éléments graphiques permettant d'expliquer facilement leur code.

Les candidats prendront garde à ne pas cacher les difficultés dans des fonctions auxiliaires non présentées. Il est par contre inutile de lister l'ensemble des fonctions présentes dans le programme (y

compris dans un diagramme d'appels qui sera en général illisible).

Une conclusion est évidemment la bienvenue, dans des proportions raisonnables : présenter les résultats de sa modélisation durant la moitié de l'oral est donc à éviter. Le jury apprécie une bonne maîtrise du vocabulaire. Nous rappelons à ce titre que la structure if n'est pas une "boucle" : en effet, elle ne permet pas d'effectuer de répétitions. Le jury préfère entendre "dans l'instruction if" voire "dans le if", plutôt que "dans la boucle if".

Le jury est très satisfait de la qualité globale des présentations des candidats.

4.4 Qualité des projets

Le thème du projet est complètement libre, ce qui permet en règle générale aux candidats de choisir un sujet qui les motive. Il est important de choisir un projet dont la difficulté soit en adéquation avec le niveau du candidat.

Cette année, les projets étaient globalement de bonne qualité et de taille raisonnable. Le nombre de lignes de code n'est pas une mesure de la qualité d'un projet. Par exemple, un projet de qualité avec 300 lignes de codes peut se voir attribuer la note maximale.

Le jury rappelle que le temps passé à travailler les graphismes et l'interface du projet est beaucoup moins rentable que celui passé à améliorer les aspects algorithmiques.

Une interface graphique via l'utilisation d'un module (tkinter, pygame, etc.) est bienvenue seulement si elle vient en complément d'un projet algorithmiquement intéressant mais ne peut pas s'y substituer. Les données d'entrée d'un programme doivent être passées en argument (ou via un fichier) et non demandées à l'utilisateur par des input, dont l'utilisation doit être réservée à des interactions avec l'utilisateur au cours du programme (par exemple pour un jeu). Nous présentons ici quelques écueils à éviter.

- Certains jeux de société ont des règles présentant un grand nombre de cas particuliers dont la traduction informatique est chronophage et ne présente pas ou peu d'intérêt. Elle engendre de nombreuses distinctions de cas, imbrications de `if`, etc.
- De manière plus générale, le code ne devrait pas faire apparaître de longues disjonctions de cas (quatre ou plus). Si de telles séries de `if`...`elif`... semblent nécessaires au candidat, c'est sans doute qu'il n'a pas suffisamment réfléchi à la structure du code.
- Lorsque le programme traite un grand nombre de données, il est pertinent de les stocker dans un fichier annexe.
- Le code doit pouvoir être exécuté sans lever d'erreur.
- On peut faire un projet informatique autour de son TIPE, mais ce projet ne doit pas se résumer à un traitement numérique de l'étude expérimentale du TIPE.

Certains projets se prêtent bien à une analyse statistique des résultats : en général, cette analyse a été faite par les candidats. Globalement, on a noté un effort des candidats pour éviter le code très répétitif (quatre fonctions différentes pour se déplacer dans les quatre directions par exemple) : ceux n'ayant pas fourni cet effort ont naturellement été pénalisés.

On peut également s'attendre à ce qu'un candidat qui étudie un problème classique se soit un peu renseigné, ne serait-ce qu'en sollicitant un moteur de recherche ou son encyclopédie préférée :

- Quelles solutions sont déjà connues ?
- Quels algorithmes utilise-t-on habituellement ?
- Quelles sont les structures de données pertinentes ?

4.5 Thèmes des projets

Cette année, les principaux thèmes abordés ont été les suivants :

- Algorithmes classiques

— algorithmes inspirés de la biologie appliqués à des problèmes classiques : algorithmes de fourmis [1], algorithmes génétiques, réseaux de neurones, etc. Nous insistons sur le fait que ces algorithmes ont pour but de résoudre un problème informatique et non de modéliser des comportements réels, même s'ils s'en inspirent.
— algorithmes pour la biologie : alignement de séquences, méthode shotgun, arbres phylogénétiques, etc.
— cryptographie : attaque contre Vigenère, etc.
  * L'analyse de la machine Enigma est technique mais pas algorithmiquement riche.
  * Les candidats doivent éviter de présenter une méthode cryptographique qu'ils ne comprennent que très partiellement.
— autres algorithmes : problèmes de graphes, labyrinthe, traitement d'image, etc.

Pour ces algorithmes, il est attendu que le candidat cite les sources qu'il a éventuellement utilisées.

- Modélisation
  Dans ce paragraphe, nous ne parlons pas des sujets décrits ci-dessus, qui adaptent des modèles biologiques à la résolution de problèmes algorithmiques ou mathématiques, mais de sujets dont le but est de modéliser l'évolution d'un système (dynamique de populations, propagation d'une maladie ou d'un feu de forêt, déplacement de poissons ou d'oiseaux, gestion d'un cheptel, ...). Même si la qualité et la pertinence de la modélisation ne sont pas les principaux éléments évalués, le jury attend du candidat qu'il ait un minimum de bon sens et qu'il y ait dans son projet matière à un développement intéressant d'un point de vue algorithmique. Ainsi, la surenchère de détails et de paramètres n'enrichit pas un modèle, mais empêche souvent l'analyse des résultats obtenus. Enfin, on attend des candidats un minimum de lucidité sur la portée et les limites de leurs modèles.
- Jeux : jeux de plateau, jeux logiques (kakuro, binaro, logimages, logipix, etc.), jeux vidéos. C'est un domaine riche et varié. Les candidats doivent cependant faire attention au choix du jeu : le traitement algorithmique peut demander un long travail de prise en compte de règles trop nombreuses, mais sans intérêt du point de vue de la programmation (succession d'instructions conditionnelles) ; dans un tel cas, le candidat peut avoir intérêt à simplifier les règles du jeu [2], pour simplifier la mise en place initiale, et garder ainsi du temps pour programmer et comparer deux intelligences artificielles, même élémentaires (la première pouvant jouer de façon très basique, voire de façon aléatoire), ou encore faire jouer une intelligence artificielle contre une de ses variantes, obtenue en modifiant certains de ses paramètres.

4.6 Style et lisibilité du code

La lisibilité du code s'est améliorée, mais les conseils des rapports précédents restent d'actualité :

- Comme demandé dans la notice du concours, le fichier python principal doit contenir quelques lignes de script permettant de tester de façon significative le projet (avec des paramètres bien choisis, pour garantir un temps d'exécution raisonnable). Cette consigne, qui n'a pas toujours été respectée, sera légèrement modifiée dans la notice 2019.
- Le code doit être raisonnablement commenté : un commentaire (*docstring*) de une ou deux lignes au début de chaque fonction expliquant ce qu'elle prend en argument, ce qu'elle fait et ce qu'elle renvoie est très appréciable.
- Dans les noms de variables, de fonctions et de fichiers, il faut éviter tous les caractères dits spéciaux, c'est-à-dire se restreindre aux lettres non accentuées (minuscules et majuscules), aux chiffres et au « tiret bas » _. Dans les chemins de fichiers il est fortement recommandé d'utiliser

1. Ces algorithmes sont pertinents pour résoudre le problème du voyageur de commerce mais pas le problème du plus court chemin.
2. Il est alors recommandé d'expliciter les simplifications choisies.

/ et non \, par exemple, on préfèrera `Donnees/exemple.txt` à `Donnees\exemple.txt`.

- Éviter d'utiliser des chemins absolus : si l'examinateur souhaite tester le programme sur sa machine, il y a peu de chance qu'un appel `f = open("D:/MonProjetInfo/Donnees/especes.txt")` fonctionne et on écrira à la place `f = open("especes.txt")` en mettant le fichier Python et le fichier de données dans le même répertoire.
  Lorsque plus d'une dizaine de fichiers sont présents, on pourra les regrouper dans un sous-dossier, et utiliser `f = open("Donnees/especes.txt")`.
  Ceci fonctionne sans problème sous Spyder, mais demande, sous Pyzo, qu'on utilise la commande *Execute file as script* (raccourci clavier Ctrl-Maj-E).
- la longueur des lignes doit être raisonnable (idéalement, pas plus de 80 colonnes). Pour faciliter cela, on pourra, dans le menu View de Pyzo, choisir une Location of long line indicator à 80 colonnes et désactiver l'option Wrap long lines. Au besoin, dans le cas où le candidat aura besoin d'écrire une très longue instruction, il pourra placer manuellement un retour à la ligne en utilisant un \.
- si l'on s'intéresse, par exemple, à l'évolution d'une population vivant dans un environnement en deux dimensions représenté par une matrice carrée, les fonctions devront être écrites pour des matrices de taille n « abstraite », ou bien en définissant au début du programme la constante n, ou bien en récupérant en début de fonction la taille de la matrice. On ne devrait ainsi jamais trouver une ligne telle que `for i in range(50)` dans le code d'une telle fonction. De manière plus générale, on essaiera toujours d'écrire un code le plus générique possible : un projet ne fonctionnant que sur un exemple précis ne présente le plus souvent que peu d'intérêt.
- de même, plutôt que d'écrire `if M[i][j] == 2` avec éventuellement un commentaire précisant « on regarde si la cellule est vivante », on préférera définir `VIVANTE = 2` au début du programme et écrire ensuite `if M[i][j] == VIVANTE`.

4.7 Notions hors-programme

Les candidats ont le droit d'utiliser des notions hors-programme (comme les dictionnaires ou les classes) ou à la limite du programme (comme les fonctions récursives). Dans certains projets, leur emploi peut être judicieux, mais un candidat utilisant une telle notion doit savoir qu'il s'expose à des questions (élémentaires) portant dessus et être capable de justifier son utilisation.

Le jury a remarqué cette année une nette amélioration à ce sujet.

4.8 Remarques spécifiques de programmation et d'algorithmique

Nous reprenons ici les remarques faites les années passées :

- Le code doit absolument être découpé en fonctions, et il faut réfléchir soigneusement au découpage le plus judicieux. Le jury note avec satisfaction que les candidats ont, cette année, bien suivi cette recommandation déjà présente dans les rapports précédents.
- De nombreux projets nécessitent de parcourir les voisins d'une case d'un tableau bidimensionnel, ce qui a été en général bien traité : les candidats savent souvent éviter de faire de nombreux cas particuliers.
- Il est fortement déconseillé d'utiliser un algorithme que l'on n'est pas capable d'expliquer correctement lorsque l'examinateur le demande. Si le projet est riche par ailleurs, le candidat peut éventuellement préciser qu'il s'est contenté de reprendre une correction de Travaux Dirigés ou que son professeur l'a aidé. Les projets très pauvres à l'exception d'une fonction relativement compliquée et non maîtrisée se sont vu attribuer des notes très basses.
- Au sujet de l'algorithme de Dijkstra, que le jury sait être un algorithme difficile à appréhender, le jury se contente, quand un candidat ne l'a pas utilisé alors qu'il aurait été bien utile, de lui demander s'il connaît un algorithme vu en cours qui aurait été adapté à son problème, mais sans

insister ensuite sur le fonctionnement de cet algorithme ; par contre, si l'algorithme de Dijkstra a été utilisé, le jury pourra vérifier que le candidat comprend l'algorithme et son code Python.

- L'algorithme de Dijkstra n'a d'intérêt que si le graphe que l'on considère est pondéré. Pour la recherche d'un plus court chemin dans un graphe non pondéré, on utilisera un parcours en largeur.
- Il est souvent possible en Python d'utiliser une boucle `for` là où une boucle `while` serait nécessaire dans certains langages (à l'aide typiquement d'un `return` dans le corps de la boucle). Les candidats sont encouragés à tirer parti de cette possibilité, mais cela ne les dispense pas de savoir écrire une boucle `while` en gérant correctement les conditions de sortie.
  L'utilisation de boucles `while` reste un point délicat pour certains candidats (confusions entre `if` et `while`).
- Plus de candidats que l'année dernière maîtrisent la différence entre une fonction renvoyant une valeur et une fonction modifiant son argument. Le jury s'en réjouit.
- Le jury a constaté des progrès sur la manipulation des booléens. Idéalement, on préfère, par exemple, lire `return x > 0` plutôt que :

```
if x > 0:
    return True
else:
    return False
```