

رياضيات بلا خوف

لمهندسي البرمجيات
بلغتي C / C++

افهم الفكرة. وثق بالشفرة.

من أعداد موثوقة إلى البيانات والذكاء الاصطناعي



30 درساً بصرياً • 4 مشروعات مشروحة خطوة بخطوة

أيمن الحراكي

رياضيات تضعها موضع التطبيق

طريق أوضح من الحسابات اليومية إلى شفرة عددية موثوقة، وتحليل البيانات، وأسس الذكاء الاصطناعي الحديث.

إعداد أيمن الحراكي

الطبعة الثانية المنقّحة • أكتوبر 2026

ما يعدك به الكتاب

ابدأ بسؤال محدد. ارسم العلاقة. احسب مثلاً صغيراً يدوياً. ثم اكتب شفرة تتوافق أنواعها ووحداتها وافترضاها مع الرياضيات. هذا هو إيقاع التعلّم في الكتاب.

تحتاج إلى خبرة برمجية اعتيادية: المتغيرات والشروط والحلقات والدوال والمصفوفات. ولا تحتاج إلى دراسة سلسلة من مقررات الرياضيات الجامعية؛ فالرموز تُقدّم حين تصبح مفيدة.

ما الذي تغيّر؟

قدّمت الطبعة الهندسية الأولى للغتّي C/C++ خريطة واسعة تضم 49 موضوعاً. وتحول هذه المراجعة مادتها إلى 30 درساً مترابطاً، وأربعة مشروعات كاملة، وأربعة أدلة تخصصية مركزة. وحلّ الاستدلال المشروح، وعقود المدخلات المعلنة، والتمارين المصحوبة بتفسير محل النصائح العامة المتكررة.

يتدرج مسار البيانات والذكاء الاصطناعي الآن عبر التحجيم، ودوال الخسارة، والتدرجات، والعصبون، والاسترجاع بالمتجهات، والانتباه، والتكميم. وتُشار التخصصات المتقدمة في مواضيعها كي لا تقطع مسار الأساسيات.

حول الأمثلة

تستخدم أمثلة C معيار C11، وأمثلة C++ معيار C++20. وتتيح مجموعات البيانات الصغيرة والثابتة فحص كل خطوة. وكل قائمة شفرة برنامج كامل. والمجالات الموصوفة وافترضات المنصة المستهدفة جزء من المثال، وليست حواشي اختيارية.

ملاحظة تحريرية: أنجزت هذه المراجعة ورسومها بمساعدة الذكاء الاصطناعي. وقد جرى تجميع الشفرة المصاحبة وتجربتها في بيئة التحقق المسجلة في الصفحة 95. تعلّم الأمثلة أسلوب الاستدلال؛ أما أنظمة الإنتاج المتخصصة فتحتاج إلى تحقق يناسب مدخلاتها ومتطلباتها.

رسالة من أيمن الحراكي

لماذا أعددت هذا الكتاب؟

أعددت هذا الكتاب في الأصل لنفسي. ففي أثناء تطوير أدوات برمجية منخفضة المستوى لمشروع MyLang Toolchain، على MyLangTC.org، احتجت إلى مراجعة مفاهيم رياضية مهمة، وفهمها بدقة، وتطبيقها بثقة على مسائل هندسية حقيقية.

وقد قطع المشروع بالفعل خطوات كبيرة. طوّرت MyLangAssembler، ولغة بسيطة جديدة تسمى «التجميع المهيكل» (Structured Assembly)، تقع بين لغة C والتجميع التقليدي. وأصممها لتكون طبقة التمثيل الوسيط في سلسلة أدواتي، بدلاً من تبني النهج الذي تقدمه LLVM IR و MLIR. ويستهدف MyLangAssembler حالياً معالجات x86-64 على Windows و Linux. وقد بلغ مرحلة ألفا، وأخطط لإتاحته قريباً للراغبين في تجربته.

قادتني حاجتي إلى أساس رياضي أوضح إلى الطبعة الأولى. قرأتها بعناية، وانتفعت بها، وقررت مشاركتها لينتفع غيري أيضاً. وتجاوز استقبالي توقعاتي؛ إذ راجعها قراء ذوو خبرة، وحددوا أخطاء، ومنحوني تشجيعاً كريماً، وساعدوني على رؤية سبل تحسينها. وأنا أقدر تصحيحاتهم ودعمهم معاً.

شجعتني استجاباتهم على إعداد هذه الطبعة الثانية بمزيد من الوضوح والتركيز، وباهتمام خاص بالبيانات والذكاء الاصطناعي من خلال C++ و C. وما يزال الهدف عملياً: أن نفهم الرياضيات بالقدر الذي يتيح استخدامها، ومساءلتها، والحكم على الشفرة التي تنفذها.

وأريد أن أوضح تماماً كيف يُنتج هذا العمل: أعتمد اعتماداً واسعاً على الذكاء الاصطناعي لاقتراح المادة، وتطوير الشروح والأمثلة، وتنظيم المحتوى، والمساعدة في تصميم الكتاب. وأصف دوري بأنه إعداد هذه المادة وجمعها ومراجعتها. ولم أدع قط أنني ألفتها مستقلاً عن هذه الأدوات. ومع ذلك، فإن اختيار المادة وربطها وفحصها وتنقيحها يتطلب جهداً كبيراً. وحين لا أكون قد راجعت كتاباً مراجعة كافية، أضع عليه وصف «مسودة».

لذلك فإن التنبيه إلى مشاركة الذكاء الاصطناعي لا يكشف شيئاً أخفيته. أرحب بالتصحيحات المحددة والنقد المتأنني؛ أما الأوصاف الانتقاصية والتكهن بدوافعي فلا تحسن معادلة أو مثالاً أو فهم قارئ. أشارك هذا العمل لتطوير مشروعاتي ومساعدة الآخرين على التعلم، لا طلباً للشهرة. ويستطيع من يرفض التعلم بمساعدة الذكاء الاصطناعي اختيار كتب أخرى، والتوقف عن متابعة حساباتي أو زيارة موقعي، دون حاجة إلى تحويل اختياره إلى عداء شخصي.

وإلى كل من يريد استكشاف الإمكانيات المدهشة لهذه الأدوات، والتعلم معي، والمساهمة في تحسين المادة: أهلاً بكم، ولكم تقديري واحترامي. هذا الكتاب لمن يريد أن يفهم ويبني ويتقدم معنا.

أيمن الحراكي

مُعِدَّ هذا الكتاب | MyLangTC.org

اقرأ. توقع. شغل. اشرح.

01 اقرأ الرسم

تبني الصفحة الأولى من كل درس تصوراً ذهنياً واحداً. اقرأ الوحدات والافتراضات قبل المعادلة.

02 توقع نتيجة المثال الصغير

اتبع الخطوات المرقمة قبل النظر إلى خرج البرنامج. فالحالة التي تستطيع التحقق منها يدوياً هي أول مرجع مستقل لك للتحقق من الصحة.

03 شغل البرنامج كاملاً

لكل درس ملف مصدر واحد في مجلد الأمثلة المصاحبة. افتح الملف المقابل وابنه باستخدام معيار اللغة الموضح أدناه.

04 غيّر شيئاً واحداً

غيّر طرف مجال، أو مقياساً، أو حجماً، أو إشارة. واشرح الأثر المتوقع قبل إعادة التشغيل، ثم ارجع إلى إجابات أسئلة التحقق في الصفحات 84-88.

بناء مثال بلغة C

```
gcc -std=c11 -O2 examples/lesson_01.c -lm -o lesson_01
```

بناء مثال بلغة C++

```
g++ -std=c++20 -O2 examples/lesson_02.cpp -o lesson_02
```

شغل البرنامج الناتج من بيئة التطوير، أو استخدم `./lesson_01` على Linux/macOS، و `lesson_01.exe` على Windows. احذف `-lm` إن كانت سلسلة أدوات المترجم لا تحتاج إلى خيار مستقل لمكتبة الرياضيات. يلزم تنفيذ متوافق مع C++20/C11 ومكتبته القياسية.

استخدام CMake أو CLion

افتح ملف `CMakeLists.txt` المصاحب. لكل درس ومشروع هدف تنفيذي مستقل، فتشغل برنامجاً كاملاً واحداً في كل مرة. ويشغل CTest فحوص الخرج واختبارات الحدود. لا حاجة إلى حزم عديدة خارجية أو مجموعات بيانات يجري تنزيلها.

يبقي ملف PDF النص والشفرة قابلين للتحديد. وللحصول على المسافات البيضاء بدقة، استخدم ملفات المصدر المصاحبة. تفترض النتائج العشرية المتوقعة بيئة `binary64` الاعتيادية المستخدمة في التحقق؛ وقد تختلف الأرقام المطبوعة الأخيرة في تطبيقات أخرى للفاصلة العائمة.

اختر فكرتك التالية

01	امنح كل عدد معنى	8
02	حوّل المعادلة إلى دالة	10
03	افهم النمو بالأسس واللوغاريتمات	12
04	افهم البتات، ثم اكتب الشرط	14
05	افحص النطاق قبل الضرب	16
06	افهم الدورات ومقاييس التمثيل	18
07	الفاصلة العائمة مسطرة متحركة	20
08	اعمل ضمن ميزانية خطأ صريحة	22
09	المجموع حلقة لها اسم	24
10	أثبت صحة حلقة في جملة	26
11	توقّع العمل قبل قياس الزمن	28
12	اكتشف البنية بالمجموعات والرسوم البيانية	30
13	عدّ النواتج قبل استخدام بايز	32
14	احتفظ بصورة توزيع البيانات	34
15	قسّ عدم اليقين، لا السرعة وحدها	36

أساس سريع

اقرأ الدروس 1-12 بالترتيب. يستخدم كل درس أفكاراً سابقة: تقود الوحدات إلى فحوص النطاق، وتقود فحوص النطاق إلى حلقات موثوقة، وتقود الحلقات إلى التعقيد الحسابي.

لكل درس صفحتان: افهم الفكرة، ثم طبّقها. وتطابق أرقام هذا الفهرس أرقام صفحات PDF الظاهرة.

اختر فكرتك التالية

38	16	بادِل الذاكرة بمعدل خطأ معلوم
40	17	المتجه قائمة ذات معنى هندسي
42	18	افحص أبعاد المصفوفات قبل القيم
44	19	أبقِ الإحداثيات ضمن أطرها
46	20	المشتقة حساسية محلية
48	21	درّب نموذجاً بتقليل الخسارة
50	22	يحوّل التراكم المعدلات إلى مجاميع
52	23	الإشارات جوارات مرّجة
54	24	عُدّ البايتات بعناية كعدّ العمليات
56	25	يبقى في العمل المتوازي جزء متسلسل
58	26	جهّز البيانات دون تسريب الإجابة
60	27	حوّل الدرجات إلى خسارة مستقرة
62	28	الانتشار العكسي إعادة استخدام لقاعدة السلسلة
64	29	يشترك التشابه والانتباه في الضرب النقطي
66	30	اجعل التكميم قراراً في ميزانية الخطأ
68		المشروع A / تقرير زمن الاستجابة
71		المشروع B / مرشّح صورة بلغة C
74		المشروع C / الانحدار
77		المشروع D / الاسترجاع بالمتجهات
80		أدلة التخصصات
84		إجابات أسئلة التحقق
89		أطلس المعادلات
92		مسرد المصطلحات
94		المراجع والملاحظات التحريرية

لغة صغيرة لأفكار كبيرة

تصبح الرموز الرياضية أسهل حين تحوّل كل رمز إلى فعل تعرفه من البرمجة. أبقِ هذه الصفحة قريبة في الدروس الأولى.

$x[i]$

اقرأ العنصر ذي الفهرس i من مجموعة مرتبة.

مجموع $x[i]$

مرّ على الفهارس المختارة واجمع القيم.

$|x|$

خذ القيمة المطلقة: تجاهل إشارة العدد الحقيقي.

$\text{sqrt}(x)$

أوجد العدد غير السالب الذي يساوي مربّعه x : ويشترط $x \geq 0$.

x^2 في الرياضيات

اضرب x في نفسه. وفي C/C++ اكتب $x*x$, لا x^2 .

$[a, b]$

أدرج الطرفين معاً.

$[a, b)$

أدرج a واستبعد b . يوافق ذلك مجالات المكررات الشائعة.

$f(x)$

طبّق قاعدة على المدخل x .

$P(A|B)$

احسب نسبة نواتج B التي تنتمي أيضاً إلى A .

dL/dw

اسأل كيف تتغير الخسارة L محلياً عندما يتغير الوزن w .

A^T

انقل المصفوفة: بدّل فهارس الصفوف والأعمدة.

يساوي تقريباً

تقريب مرتبط بمقياس خطأ أو بنموذج معن، وليس تطابقاً تاماً.

امنح كل عدد معنى

يصبح العدد مفيداً حين تعرف ما الذي يقيسه.

قبل تخصيص مخزن مؤقت أو تقدير عملية نقل، اكتب الوحدات. يكشف ذلك أخطاء لا يستطيع المترجم رؤيتها. للكمية ثلاثة أجزاء: قيمة، ووحدة، ودور. فقد يعني العدد 48,000 إطارات صوتية في الثانية، أو بايتات، أو عينات. وهذه المعاني ليست قابلة للاستبدال بعضها ببعض.

تقارن النسبة بين كميتين؛ أما المعدل فيقسم نوعاً من الكميات على نوع آخر. فقسمة البايتات على البايتات بلا وحدة، وقسمتها على الثواني تعطي معدل نقل.

يعني MB العشري 1,000,000 بايت، ويعني MiB الثنائي 1,048,576 بايت. والبايت في C هو sizeof(char)، ولا يتألف دائماً من ثمانية بتات. تفترض المنصات الشائعة في المثال الصوتي بايتات من ثمانية بتات، ونوع float بحجم أربعة بايتات.

$$48,000 \times 2 \times 4 = 384,000$$

إطارات

عينات / إطار

بايتات / عينة

بايتات

تُختزل الوحدات، ويظل المعنى واضحاً.

احسبها خطوة بخطوة

1. تضم ثمانية واحدة من صوت ستيريو 48,000 إطار، في كل إطار عينتان.
2. تشغل كل عينة أربعة بايتات: $48,000 \times 2 \times 4 = 384,000$ بايت.
3. $384,000 / 1,048,576 = 0.3662109375$ MiB. يصف العدد الذي يبدو أصغر الذاكرة نفسها.

الإطارات × عينات / إطار × بايتات / عينة = بايتات

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_01.c

```
#include <limits.h>
#include <stdio.h>
int main(void) {
    _Static_assert(CHAR_BIT == 8, "8-bit bytes required");
    _Static_assert(sizeof(float) == 4, "4-byte float required");
    const size_t frames = 48000;
    const size_t channels = 2;
    const size_t bytes = frames * channels * sizeof(float);
    printf("%zu bytes\n", bytes);
    printf("%.9f MiB\n", (double)bytes / 1048576.0);
    return 0;
}
```

الناتج المتوقع

384000 bytes
0.366210938 MiB

يجعل تحويل النوع القسمةً بفاصلة عائمة. والثوابت صغيرة بما يكفي لهذه المنصة؛ أما الأحجام العامة فتحتاج إلى الضرب المفحوص في الدرس 5.

انتبه إلى الحدود

الانتقال من 100 ms إلى 80 ms يعني تقليل الزمن بنسبة 20%، لكن معدل الإنجاز يرتفع بنسبة 25% إذا ثبت العمل في كل تشغيل. سمِّ المقام دائماً.

جرب قبل قراءة الإجابة

يحتوي ملف على 750 MiB. كم ثانية يستغرق نقل حمولته عند معدل مثالي قدره 2.5 Gbit/s؟ تجاهل جميع أعباء البروتوكولات.

الإجابة 01 • الصفحة 84

حوّل المعادلة إلى دالة

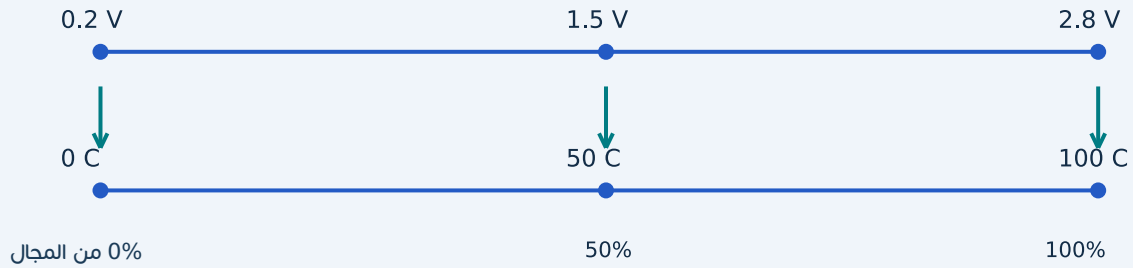
المعادلة برنامج صغير له عقد واضح.

تربط المعايير، ومنزلقات الواجهات، وتحجيم السمات مجالاً بمجال آخر. اشتق التحويل من الطرفين.

للدالة مجال مدخلات ومدى مخرجات. حددهما قبل اختيار النوع. فقد لا تقبل معايرة المستشعر إلا مجال جهد محدداً.

حوّل المدخل أولاً إلى نسبة من مجاله، ثم استخدم هذه النسبة للانتقال بين طرفي الخرج. يفترض الاستيفاء الخطي علاقة على خط مستقيم.

يغيّر الحصر عند الحدود (Clamping) العقد: إذ يستبدل القيم الخارجة عن المجال بقيم طرفيه. وهذا مفيد لمنزلق، لكنه قد يخفي عطل مستشعر. لذلك يرفض المثال المدخل غير الصالح.



احسبها خطوة بخطوة

1. يعطي مستشعر 0.2 V عند 0 °C و 2.8 V عند 100 °C.
2. عند 1.5 V تكون النسبة $0.5 = (1.5 - 0.2) / (2.8 - 0.2)$.
3. يكون الخرج $50\text{ °C} = 0 + 0.5 \times (100 - 0)$. تحقق من الطرفين أيضاً، لا من نقطة المنتصف وحدها.

$$t = (x - a) / (b - a); y = \text{low} + t(\text{high} - \text{low})$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_02.cpp

```
#include <cmath>
#include <iostream>
#include <stdexcept>
double celsius(double volts) {
    if (!std::isfinite(volts) || volts < 0.2 || volts > 2.8)
        throw std::domain_error("volts outside calibration");
    const double t = (volts - 0.2) / (2.8 - 0.2);
    return std::lerp(0.0, 100.0, t);
}
int main() {
    std::cout << celsius(1.5) << '\n';
}
```

الناتج المتوقع

50

تعبّر `std::lerp` في C++20 عن الاستيفاء مباشرة. ولتحويل عام، ارفض تساوي طرفي المجال الأصلي، وحدد السلوك عند عكس ترتيبهما.

انتبه إلى الحدود

فحص المساواة التامة مناسب لطرفين متساويين تماماً. أما المجال الذي يقترب عرضه من الصفر فيحتاج إلى قاعدة مستقلة لحساسية المسألة، مبنية على الوحدات؛ ولا تمثل قيمة إبسيلون عامة حلاً.

جرب قبل قراءة الإجابة

حوّل $x = 25$ من المجال $[10, 70]$ إلى $[1000, 4000]$. ماذا يحدث إن أعطى المستدعي $x = 80$ ؟

الإجابة 02 • الصفحة 84

افهم النمو بالأسس واللوغاريتمات

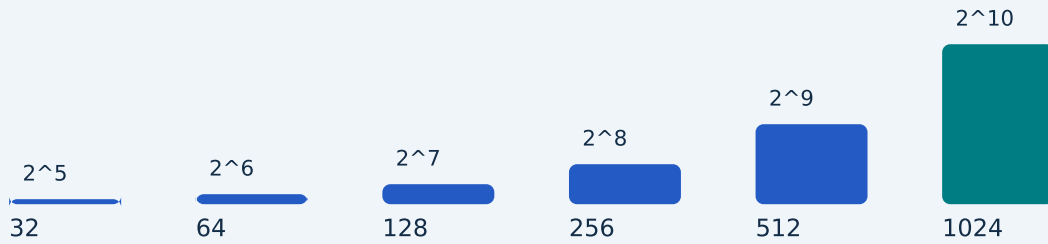
يسأل اللوغاريتم: كم مرة ضرب؟

تعدّ الأسس الحالات. وتستعيد اللوغاريتمات عدد القرارات أو المضاعفات أو البتات اللازمة للوصول إليها. يعطي خياران لكل بت 2^b نمطاً ممكناً من البتات. وتميّز عشرة بتات 1,024 نمطاً، بقيم غير موقعة من 0 إلى 1,023.

تكون $\log_2(1024) = 10$ لأن $2^{10} = 1024$. ولا تحتاج إلى لوغاريتم بفاصلة عائمة لعدّ بتات عدد صحيح.

لعدد N من الحالات المميزة، المرقمة من 0 إلى $N-1$ ، يكون عدد البتات رياضياً $\lceil \log_2(N) \rceil$ عندما يكون N موجباً. وتحتاج الحالة الوحيدة إلى صفر من بتات المعلومات، وإن ظل الكائن المخزن يشغل حيزاً.

تضاعف السعة في كل خطوة.



احسبها خطوة بخطوة

1. يحتاج جدول إلى سعة تكفي 1,000 عنصر. وقوتا الاثنين المجاورتان هما 512 و 1,024.
2. السعة 1,024 هي أول سعة تكفي.
3. قبل مضاعفة عدد صحيح، تحقّق من إمكان تمثيل القيمة التالية. والإزاحة بمقدار عرض النوع ليست اختصاراً صالحاً.

$$2^{10} = 1024; \log_2(1024) = 10$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_03.cpp

```
#include <cstdint>
#include <iostream>
#include <limits>
#include <optional>
std::optional<std::size_t> next_power(std::size_t n) {
    std::size_t p = 1;
    while (p < n) {
        if (p > std::numeric_limits<std::size_t>::max() / 2)
            return std::nullopt;
        p *= 2;
    }
    return p; // policy: zero requests get capacity one
}
int main() {
    const auto p = next_power(1000);
    if (p) std::cout << *p << '\n';
}
```

الناتج المتوقع

1024

لا يتجاوز عدد دورات هذه الحلقة عددَ بتات القيمة في `size_t`. وهي تُظهر فحص الفيض بوضوح؛ وتوفر أدوات البتات في C++20 بدائل لها شروطها المسبقة.

انتبه إلى الحدود

يعني الرمز `^` في C و C++ عملية XOR على البتات، وليس الرفع إلى قوة. استخدم الضرب لقوة صحيحة صغيرة ودقيقة، ودالة رياضية مناسبة للأس الحقيقي.

جرب قبل قراءة الإجابة

كم بتاً يميّز مليون معرّف؟ وما أصغر سعة من قوى الاثنين لا تقل عن 5,000؟

الإجابة 03 • الصفحة 84

افهم البتات، ثم اكتب الشرط

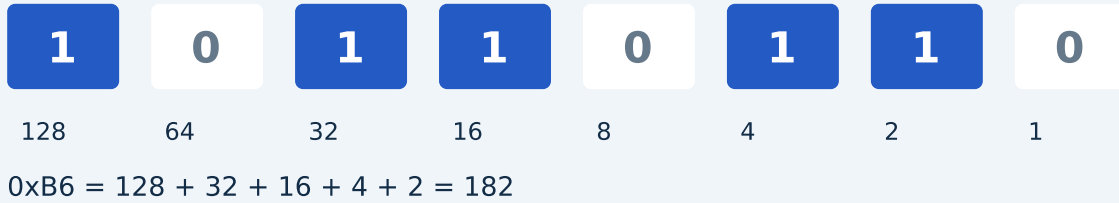
يعمل المنطق البتّي على المواضع، ويعمل المنطق البولياني على الصدق.

تصبح الأعلام والشروط أسهل مراجعة حين تصوغ القاعدة المقصودة في جملة أولاً.

العدد الثنائي مجموع قوى لل اثنين. ويجمع النظام الست عشري كل أربعة بتات في رقم واحد. فالعدد 0xB6 هو 1011 0110، أي $128 + 32 + 16 + 4 + 2 = 182$.

تختار AND البتّية البتات المضبوطة المشتركة. وتجمع OR البتات المضبوطة، وتحدد XOR الاختلافات. أما && و|| المنطقيتان فتجمعان قيم الصدق وتستخدمان التقييم المختصر.

تقدم قاعدة دي مورغان وسيلة مفيدة للقراءة: نفي «A وB» يعني «نفي A أو نفي B». أبقِ الشروط خالية من الآثار الجانبية حين تعيد ترتيبها.



احسبها خطوة بخطوة

1. قد لا يُسمح بكتابة ملف إلا إذا كان المستخدم موثّق الهوية، وكان مالك الملف أو مسؤولاً.
2. اكتب (admin || owner) && authenticated. توضّح الأقواس تجميع الشروط.
3. لثلاثة مدخلات بوليانية ثماني توليفات. وفحصها جميعاً اختبار صغير ومكتمل لجدول الحقيقة.

$$\text{NOT (A AND B) = (NOT A) OR (NOT B)}$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_04.c

```
#include <stdbool.h>
#include <stdint.h>
#include <stdio.h>
bool can_write(bool auth, bool owner, bool admin) {
    return auth && (owner || admin);
}
int main(void) {
    const uint32_t read_flag = UINT32_C(1) << 1;
    const uint32_t write_flag = UINT32_C(1) << 3;
    const uint32_t flags = read_flag | write_flag;
    printf("write flag: %d\n", (flags & write_flag) != 0);
    printf("allowed: %d\n", can_write(true, false, true));
    return 0;
}
```

الناتج المتوقع

```
write flag: 1
allowed: 1
```

مقادير الإزاحة ثابتة وصالحة. ولا تتوافر الأنواع ذات العرض الثابت، مثل `uint32_t`، إلا في التطبيقات التي توفر النوع المقابل بعرضه الدقيق.

انتبه إلى الحدود

استخدم `*p > 0 && p != NULL` لفحص المؤشر بالتقييم المختصر. استبدال `&&` بالرمز `&` يقيّم المعاملين معاً ويزيل هذه الحماية.

جرب قبل قراءة الإجابة

اكتب `0xA7` بالثنائي. وأعد كتابة `((local || cached) && ready)` دون نفي يحيط بتعبير مركب.

الإجابة 04 • الصفحة 84

افحص النطاق قبل الضرب

قد تكون الإجابة الرياضية صحيحة، لكنها لا تتسع في نوع الآلة.

تحتاج أحجام التخصيص وأبعاد الصور والعدادات إلى تبرير للنطاق، فضلاً عن المعادلة.

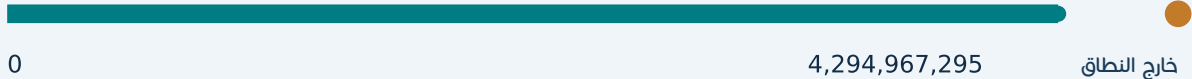
يلتف الحساب غير الموقع بترديد يزيد واحداً على أكبر قيمة له. يفيد ذلك بعض العدادات، لكنه خطر عند حساب عدد البايتات. أما فيض الأعداد الصحيحة الموقعة فسلوكه غير معرّف.

يأتي تحويل النوع بعد العملية متأخراً عن توسيع نوع النتيجة الوسيطة. وسّع النوع قبل الضرب، ثم اسأل أيضاً إن كان حاصل الضرب يتسع في النوع الأوسع.

للأعداد الصحيحة غير السالبة، يتسع $a*b$ ضمن الحد الأقصى M إذا وفقط إذا كان a صفراً أو تحقق $b \leq M/a$. يحوّل ذلك ضرباً خطراً إلى مقارنة آمنة.

نطاق نوع غير موقع من 32 بتاً

5 مليارات



افحص المعاملين قبل حساب حاصل الضرب.

احسبها خطوة بخطوة

1. يستوعب نوع غير موقع من 32 بتاً قيماً حتى 4,294,967,295.
2. حاصل $100,000 \times 50,000$ هو 5,000,000,000، ولذلك لا يتسع.
3. يتسع على منصة يكون فيها `size_t` من 64 بتاً؛ لكن ينبغي للواجهة نفسها أن ترفض أيضاً المدخلات التي يتجاوز حاصلها الحد الفعلي لتلك المنصة.

يتسع $a*b$ إذا كان $a == 0$ أو $b \leq MAX/a$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_05.c

```
#include <stdbool.h>
#include <stdint.h>
#include <stdio.h>
bool size_product(size_t a, size_t b, size_t *out) {
    if (out == NULL || (a != 0 && b > SIZE_MAX / a))
        return false;
    *out = a * b;
    return true;
}
int main(void) {
    size_t bytes = 0;
    if (!size_product(48000, 8, &bytes)) return 1;
    printf("%zu\n", bytes);
    printf("rejected: %d\n", !size_product(SIZE_MAX, 2, &bytes));
    return 0;
}
```

الناتج المتوقع

```
384000
rejected: 1
```

يُفحص مؤشر الخرج، ولا تعدّل الاستدعاءات الفاشلة النتيجة. وحساب حاصل ضرب ثلاثة أبعاد يحتاج إلى عمليتي ضرب مفحوصتين.

انتبه إلى الحدود

قد ينتج التحويل غير المفحوص من `int` سالب إلى `size_t` قيمة هائلة. تحقق من المدخل الموقع قبل تحويله. وقد تخضع المعاملات الصغيرة غير الموقعة للترقية الصحيحة (Integer Promotion).

جرب قبل قراءة الإجابة

حدّد شروطاً مسبقة آمنة لجمع قيمتين من `size_t`، ولتقريب عدد صحيح `x` صعوداً إلى محاذاة `A`.

الإجابة 05 • الصفحة 84

افهم الدورات ومقاييس التمثيل

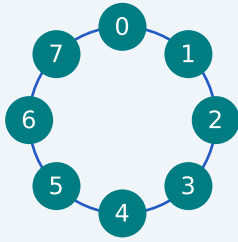
يصف الباقي دورة، ويصف المقياس كسراً مخزناً.

تبدو المخازن الدائرية وقيم الفاصلة الثابتة أعداداً صحيحة عادية، لكن لكل منها معنى رياضي مختلف.

في حلقة سعتها 8 تكون المواضيع من 0 إلى 7. والموضع التالي بعد 7 هو 0. تعبّر عملية بتريد عن هذا الالتفاف، بشرط أن تكون السعة موجبة.

يتبع باقي القسمة في C قسمة تُقَرَّب نحو الصفر: فالناتج 8 % -1 هو -1. ولمعامل موجب m، احسب $r = x \% m$ ثم أضف m فقط إذا كان r سالباً، لتحصل على باقي غير سالب.

يمثل العدد الخام q ذو F بتات كسرية في الفاصلة الثابتة القيمة $q / 2^F$. فالقيمة الخام 384 في Q8.8 تعني 1.5. ويحتاج الضرب إلى نتيجة وسيطة أوسع، وتقريب صريح، وفحص نطاق الخرج.



7 -> 0

يلتف الموضع؛ وتظل السعة 8.

يصف الحساب بتريد دورة من الحالات.

احسبها خطوة بخطوة

1. تتكرر المرحلة المشتركة لمهمة كل 12 ms وأخرى كل 18 ms بعد 36 ms لأول مرة.
2. تعطي خوارزمية إقليدس $\text{gcd}(12, 18) = 6$. ثم $\text{lcm} = (12/6) \times 18 = 36$.
3. اقسم قبل الضرب لتقليل الخطر؛ ومع الفترات العامة يظل حاصل الضرب المتبقي محتاجاً إلى فحص النطاق.

$\text{gcd}(a, b) = \text{gcd}(b, a \bmod b)$; $\text{real} = \text{raw} / 2^F$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_06.c

```
#include <stddef.h>
#include <stdio.h>
size_t next_ring(size_t i, size_t capacity) {
    /* Precondition: capacity > 0 and i < capacity. */
    return i == capacity - 1 ? 0 : i + 1;
}
unsigned gcd_u(unsigned a, unsigned b) {
    while (b != 0) {
        unsigned r = a % b;
        a = b; b = r;
    }
    return a;
}
int main(void) {
    printf("next: %zu\n", next_ring(7, 8));
    printf("gcd: %u\n", gcd_u(12, 18));
    printf("scaled: %.1f\n", 384 / 256.0);
    return 0;
}
```

الناتج المتوقع

```
next: 0
gcd: 6
scaled: 1.5
```

تتجنب دالة الحلقة حساب $i+1$ الذي قد يفيض عند الحد. شرطها المسبق صريح؛ وتفترض هذه الدالة الداخلية الصغيرة أن المستدعي تحقق منه.

انتبه إلى الحدود

لا يحل القناع محل العملية بترديد إلا إذا كانت السعة قوةً لل اثنين. كذلك لا تمثل إزاحة قيمة موقعة سالبة يميناً سياسة تقريب محمولة في C11؛ أما القسمة فتُقَرَّب نحو الصفر.

جرب قبل قراءة الإجابة

احسب $\text{gcd}(252, 105)$. ولعدّد نبضات يلتف بعرض 8 بتات، ما الزمن المنقضي الذي يمثل 4-250 إن كان قد مر أقل من 256 نبضة؟

الإجابة 06 • الصفحة 84

الفاصلة العائمة مسطرة متحركة

يتغير التباعد بين قيم الفاصلة العائمة مع مقدار العدد.

يشرح فهم التمثيل كثيراً من النتائج المفاجئة دون حفظ حالات خاصة.

تخزن قيمة الفاصلة العائمة إشارةً، وجزءاً دالاً (Significand)، وأساساً. تخيل مسطرة تتباعد علاماتها كلما ازداد الأس.

في منصات IEEE binary64 الشائعة، يملك double دقة جزء دال من 53 بتاً. ويمكن تمثيل كل عدد صحيح حتى 2^{53} تمثيلاً دقيقاً؛ وبعده يتعذر تمثيل بعض الأعداد الصحيحة المتجاورة.

الكسر العشري 0.1 دوري في الثنائي، كما أن $1/3$ دوري في العشري. ولا مفر من التقريب. تتوافر اللانهاية NaN في تطبيقات IEEE؛ اختبر منصتك بدل افتراض أن جميع تطبيقات C أو C++ تستخدم الصيغة نفسها.

قرب مقدار أصغر



عند ضعف المقدار



تباعد توضيحي: تزداد الفجوات مع الأس.

احسبها خطوة بخطوة

1. تُقَرَّب كل من 0.1 و 0.2 عند التخزين.
2. يُقَرَّب مجموعهما مرة أخرى. ويكون عادة 0.30000000000000004 في binary64.
3. تختلف عنه القيمة 0.3 التي قُرِّبت بصورة مستقلة. هذا أثر للتمثيل، وليس خلافاً في عامل الجمع.

التباعد المحلي قرب 1 في binary64 يساوي 2^{-52}

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_07.cpp

```
#include <cmath>
#include <iomanip>
#include <iostream>
#include <limits>
int main() {
    static_assert(std::numeric_limits<double>::is_iec559);
    static_assert(std::numeric_limits<double>::digits == 53);
    const double x = 0.1 + 0.2;
    const double after = std::nextafter(1.0, 2.0);
    std::cout << std::setprecision(17) << x << '\n';
    std::cout << "equal: " << (x == 0.3) << '\n';
    std::cout << "spacing: " << after - 1.0 << '\n';
}
```

الناتج المتوقع

```
0.300000000000000004
equal: 0
spacing: 2.2204460492503131e-16
```

تجعل التأكيدات الساكنة التمثيل المستخدم في العرض شرطاً للبناء. وتطلب nextafter القيمة التالية القابلة للتمثيل في اتجاه معاملها الثاني.

انتبه إلى الحدود

إبسيلون هي الفجوة قرب الواحد، وليست سماحية قياس عامة. تحتاج النقود والطوابع الزمنية وقراءات المستشعرات المشوشة إلى عقود خطأ مختلفة.

جرب قبل قراءة الإجابة

هل يستطيع double بصيغة binary64 التمييز بين العددين الصحيحين 9,007,199,254,740,992 و 9,007,199,254,740,993؟

الإجابة 07 • الصفحة 85

اعمل ضمن ميزانية خطأ صريحة

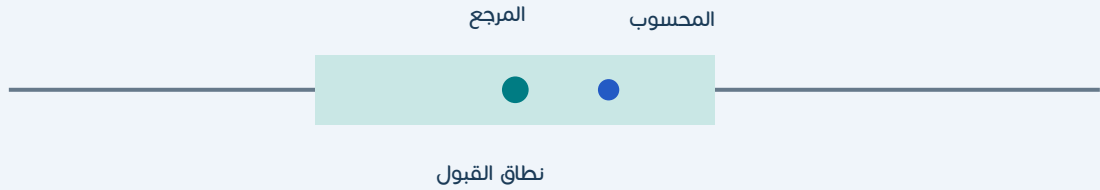
اختر الدقة من متطلبات المسألة، ثم اختر طريقة الحساب.

تضبط الشفرة العددية الجيدة الخطأ بدل أن تطلب دقة تامة مستحيلة من كل حساب بقيم حقيقية.

يقيس الخطأ المطلق المسافة بوحدة الخرج. ويقسم الخطأ النسبي على مقدار مرجع غير صفري. وقرب الصفر لا يفيد الخطأ النسبي وحده.

تصف حساسية المسألة (Conditioning) تأثير الجواب بمدخلاته. ويصف الاستقرار مقدار الخطأ الإضافي الذي تدخله الخوارزمية. ولا تستطيع خوارزمية مستقرة استعادة معلومات تفتقدها مدخلات غير مؤكدة.

قد يكشف الطرح خطأ التقريب عند تلاشي قيمتين متقاربتين. وللقيم الصغيرة من x تحسب $\log_{10}(x)$ و $\expm1(x)$ التعبيرين $\log(1+x)$ و $\exp(x)-1$ دون فقد الزيادات الضئيلة أولاً.



احسبها خطوة بخطوة

1. قراءة 100.02°C قياساً إلى 100.00°C لها خطأ مطلق 0.02°C .
2. خطؤها النسبي 0.0002 ، أي 0.02% . وقد يهم أي منهما بحسب الموصفة.
3. تقبل قاعدة مركبة فرقاً ضمن أكبر القيمتين: السماحية المطلقة، والسماحية النسبية مضروبة في المقياس. حدّد السماحيات بوحدات فيزيائية وكسور ذات معنى.

$$|a-b| \leq \max(\text{abs_tol}, \text{rel_tol} \times \max(|a|, |b|))$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_08.cpp

```
#include <algorithm>
#include <cmath>
#include <iostream>
bool near(double a, double b, double at, double rt) {
    if (!std::isfinite(a) || !std::isfinite(b) ||
        !std::isfinite(at) || !std::isfinite(rt) ||
        at < 0 || rt < 0) return false;
    const double scale = std::max(std::abs(a), std::abs(b));
    if (scale == 0) return true;
    // Scale first: avoid overflowing a-b for opposite signs.
    return std::abs(a/scale - b/scale) <=
        std::max(at/scale, rt);
}
int main() {
    std::cout << near(0.1 + 0.2, 0.3, 1e-12, 1e-12) << '\n';
}
```

الناتج المتوقع

1

لا تقبل هذه المقارنة إلا قيماً منتهية وسماحيات منتهية غير سالبة. وهي ترفض اللانهايات وNaNs عمداً. إنها اختبار عددي تقريبي، وليست حساباً دقيقاً عند حد السماحية.

انتبه إلى الحدود

لا يلزم أن تكون المساواة التقريبية متعدية. بسماحية 0.1، يكون 0 قريباً من 0.09، و0.09 قريباً من 0.18، لكن 0 ليس قريباً من 0.18. فلا تستخدمها لمساواة مفاتيح الحاويات.

جرب قبل قراءة الإجابة

أيهما أفضل عادة قرب $x = 0$: $\log(1+x)$ أم $\log_{1p}(x)$ ؟ وهل تصلح زيادة الدقة مدخلاً قيس بصورة سيئة؟

الإجابة 08 • الصفحة 85

المجموع حلقة لها اسم

ترميز سيغما جمع متكرر، وليس عملية جديدة.

تصبح المجاميع الجارية واستعلامات المجالات بسيطة حين يكون اصطلاح الحدود واضحاً.

إن مجموع $x[i]$ عندما يمتد i من 0 إلى $n-1$ هو تماماً ما تحسبه حلقة بشرط $i < n$. والترميز يخفي أعمال التتبع فحسب.

تبدأ مصفوفة المجاميع البادئة P بالقيمة $P[0] = 0$. وتخزن $P[k]$ مجموع أول k عناصر. ومن ثم يكون مجموع المجال نصف المفتوح $[l, r)$ هو $P[r] - P[l]$.

يُؤجل مجمّع صحيح أوسع حدوث الفيض، لكنه لا يلغي إمكانه. وتتأثر مجاميع الفاصلة العائمة أيضاً بترتيب الجمع؛ وقد يقلل الجمع الزوجي أو المعوّض الخطأ في شروط مناسبة.

x	3	1	4	2	
P	0	3	4	8	10

$$P[3] - P[1] = 8 - 3 = 5$$

احسبها خطوة بخطوة

1. للقيم $x = [3, 1, 4, 2]$ ، اكتب $P = [0, 3, 4, 8, 10]$.
2. يضم المجال $[1, 3)$ القيمتين 1 و4. ومجموعه $P[3] - P[1] = 8 - 3 = 5$.
3. ابن المصفوفة مرة واحدة بكلفة $O(n)$ ، ثم أجب عن كل استعلام مجال صالح بكلفة $O(1)$. التخزين الإضافي هو $n+1$ من المجاميع.

$$P[0] = 0; P[k+1] = P[k] + x[k]$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_09.c

```
#include <stdio.h>
int main(void) {
    const int x[4] = {3, 1, 4, 2};
    long long p[5] = {0};
    for (size_t i = 0; i < 4; ++i)
        p[i + 1] = p[i] + x[i];
    const size_t left = 1, right = 3;
    printf("range sum: %lld\n", p[right] - p[left]);
    return 0;
}
```

الناتج المتوقع

range sum: 5

تبقى البيانات الثابتة كل مجموع صغيراً. ويجب أن تفحص النسخة العامة $n \geq r \geq l$ ، وأن تضمن اتساع كل مجموع متراكم وكل عملية طرح.

انتبه إلى الحدود

لا تبيح الهوية $1 + \dots + n = n(n+1)/2$ نتيجة وسيطة تفيض. اقسّم عاملاً زوجياً أولاً، ثم افحص الضرب أيضاً.

جرب قبل قراءة الإجابة

باستخدام مصفوفة المجاميع البادئة نفسها، احسب $(0,4)$ و $(2,2)$ و $(2,4)$. ماذا يعني المجال الفارغ؟

الإجابة 09 • الصفحة 85

أثبت صحة حلقة في جملة

يصف ثابت الحلقة ما يظل صحيحاً بينما تتقدم الحلقة.

غالباً ما يمنع ثابت قصير أخطاء أكثر من قائمة طويلة من رقع الحالات الخاصة.

يحتاج البرهان إلى حقيقة بدء، وخطوة تحفظها، وسبب للتوقف. في مسح القيمة العظمى، تكون الحقيقة: best هي القيمة العظمى للبادئة التي جرت زيارتها.

في البحث عن الحد الأدنى (Lower Bound)، أبقِ $[lo, hi)$ منطقة لم يُحسم أمرها. كل عنصر قبلها أصغر من المفتاح، وكل عنصر عند hi أو بعدها لا يقل عن المفتاح.

يتقلص المجال في كل دورة. وحين يفرغ، يكون lo أول موضع لا تقل قيمته عن المفتاح، أو n إن لم توجد قيمة كهذه.



احسبها خطوة بخطوة

1. ابحث عن 4 في $[2, 4, 4, 9]$. ابدأ بـ $lo=0$, $hi=4$.
2. في $mid=2$ توجد القيمة 4، إذن $hi=2$. ثم توجد 4 أيضاً في $mid=1$ ، إذن $hi=1$.
3. في $mid=0$ توجد 2، إذن $lo=1$. تنتهي الحلقة عند الفهرس 1، أي أول قيمة 4.

```
0 <= lo <= hi <= n; mid = lo + (hi-lo)/2
```

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_10.cpp

```
#include <cstdint>
#include <iostream>
#include <span>
std::size_t lower(std::span<const int> a, int key) {
    std::size_t lo = 0, hi = a.size();
    while (lo < hi) {
        const std::size_t mid = lo + (hi - lo) / 2;
        if (a[mid] < key) lo = mid + 1;
        else hi = mid;
    }
    return lo;
}
int main() {
    const int a[] = {2, 4, 4, 9};
    std::cout << lower(a, 4) << '\n';
}
```

الناتج المتوقع

1

الشرط المسبق هو ترتيب المدخل تصاعدياً مع السماح بالتساوي. وتتجنب صيغة المنتصف جمع الطرفين. وتعيد المصفوفات الفارغة صفراً بصورة طبيعية دون فك إشارة أي مؤشر.

انتبه إلى الحدود

تعرض الاختبارات حالات مختارة؛ ولا تثبت ادعاءً عاماً. ويظل البرهان على الأعداد الصحيحة الرياضية ناقصاً إن أمكن للتنفيذ أن يفيض أو يصل إلى كائن انتهى عمره.

جرب قبل قراءة الإجابة

ماذا ينبغي أن تعيد lower للمفاتيح 1 و 5 و 10 في المصفوفة نفسها؟ صغ ثابتاً لمسح القيمة العظمى.

الإجابة 10 • الصفحة 85

توقع العمل قبل قياس الزمن

عدّ بنية العمل، ثم قس كلفته.

يساعد التعقيد في اختيار خوارزمية، لكنه لا يتنبأ بعدد دقيق من المللي ثواني.

يزور المسح الخطي n عنصراً. وتجربة كل زوج غير مرتب تزور $n(n-1)/2$ زوجاً. وينشئ اثنا عشر مفتاحاً مستقلاً 2^{12} إعداداً. $= 4,096$

تمثل O حداً علوياً تقاربياً، وتمثل Θ (Theta) حدين علوياً وسفلياً متطابقين في الرتبة. وأسوأ حالة، والقيمة المتوقعة، والكلفة المطفأة أوصاف مختلفة وليست مترادفات.

للترتيب بالدمج نحو $\log_2(n)$ مستوى، بكلفة دمج $O(n)$ في كل مستوى. ولشجرة بحث متوازنة ارتفاع لوغاريتمي؛ وقد يكون ارتفاع شجرة بحث ثنائية سيئة الشكل خطياً. والكومة (Heap) مرتبة جزئياً، وليست مرتبة بالكامل.

أزواج غير مرتبة	زيارات خطية	حجم المدخل n
499,500	1,000	1,000
49,995,000	10,000	10,000

مضاعفة المدخل 10 مرات تعني نحو 100 ضعف من الأزواج.

احسبها خطوة بخطوة

1. عندما $n=1,000$ يكون عدد كل الأزواج غير المرتبة 499,500.
2. عندما $n=10,000$ يصبح العدد 49,995,000، أي نحو مئة ضعف.
3. لا ينمو المسح إلا عشرة أضعاف. وقد يتفوق تغيير الخوارزمية بكثير على تحسين عملية كل زوج.

عدد الأزواج غير المرتبة: $n(n-1)/2$ ؛ كلفة الدمج: $\sim n \log_2(n)$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_11.c

```
#include <stdio.h>
unsigned levels(unsigned n) {
    unsigned count = 0;
    while (n > 1) {
        n = n / 2 + n % 2; /* ceiling division by 2 */
        ++count;
    }
    return count;
}
int main(void) {
    printf("levels for 1000: %u\n", levels(1000));
    printf("pairs for 1000: %llu\n", 1000ULL * 999ULL / 2);
    return 0;
}
```

الناتج المتوقع

```
levels for 1000: 10
pairs for 1000: 499500
```

تقرّب $n/2 + n\%2$ إلى الأعلى دون تعريض $n+1$ للفيض. ومثال الأزواج حساب محدود، وليس تنفيذاً عاماً للتوافقيات.

انتبه إلى الحدود

يمنح النمو الهندسي لـ vector الإلحاق كلفة مطفأة ثابتة، لأن مجموع العناصر المنقولة متسلسلة هندسية. ولا يعني ذلك أن كل عنصر منفرد يُنقل مرة واحدة فقط.

جرب قبل قراءة الإجابة

اختر 3 مرمّات مختلفة من 10. كم اختياراً يوجد إن لم يكن الترتيب مهماً؟ وكم اختياراً إن كان الترتيب يحدد خط معالجة؟

الإجابة 11 • الصفحة 85

اكشف البنية بالمجموعات والرسوم البيانية

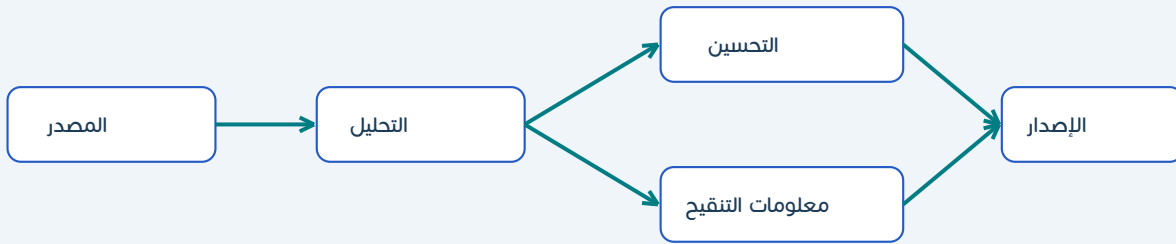
تجيب المجموعة عن الانتماء، ويجب الرسم البياني عن الاتصال.

تستخدم أدوات الاعتماديات وخطوط معالجة البيانات والمترجمات العائلة الصغيرة نفسها من البنى.

يجمع الاتحاد الأعضاء، ويبقي التقاطع الأعضاء المشتركين، ويحذف الفرق الأعضاء. ويجب أن تنتج المفاتيح المتساوية في حاوية تجزئة قيم تجزئة متساوية.

يتكون الرسم البياني من رؤوس وحواف. والاتجاه مهم: فقد يعني $B \rightarrow A$ أن B تعتمد على A . صرّح بهذا الاصطلاح، لأن عكسه يعكس ترتيب البناء.

يجد البحث بالعرض (BFS) أقصر عدد حواف في رسم غير موزون. وتحتاج خوارزمية ديكسترا إلى أوزان غير سالبة. وللرسم الموجّه الخالي من الدورات ترتيب طوبولوجي؛ وتمنع الدورة الموجّهة وجوده.



احسبها خطوة بخطوة

1. لتكن $A = \{1, 2, 3\}$ و $B = \{2, 3, 4\}$. تقاطعهما $\{2, 3\}$ ، واتحادهما $\{1, 2, 3, 4\}$.
2. في رسم الاعتماديات الموضح، يجب أن ينتهي التحليل قبل التحسين، والتحسين قبل الإصدار.
3. باستخدام قوائم التجاور، تكون كلفة BFS كامل $O(V+E)$. وتستخدم المصفوفة الكثيفة V^2 خانة، حتى لو كان عدد الحواف قليلاً.

عمل BFS: $O(V + E)$ ؛ التخزين الكثيف: $O(V^2)$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_12.cpp

```
#include <algorithm>
#include <iostream>
#include <iterator>
#include <set>
int main() {
    const std::set<int> a{1, 2, 3}, b{2, 3, 4};
    std::set<int> common;
    std::set_intersection(a.begin(), a.end(),
                          b.begin(), b.end(),
                          std::inserter(common, common.end()));
    for (int x : common) std::cout << x << ' ';
    std::cout << '\n';
}
```

الناتج المتوقع

2 3

تحتاج `set_intersection` إلى مجالات مرتبة بترتيب متوافق. وتوفر `std::set` ذلك هنا. أما ترتيب المرور على مجموعة تجزئة فلا يحقق هذا الشرط المسبق.

انتبه إلى الحدود

تحتاج الحاوية المرتبة إلى ترتيب ضعيف صارم. وقد تنتهك هذا العقد مقارنات تعتمد على حالة متغيرة، أو سماعات غير متسقة، أو قيم NaN غير معالجة.

جرب قبل قراءة الإجابة

يضم رسم بياني مليون رأس وثلاثة ملايين حافة. كم بايتاً تحتاج مصفوفة تجاور تستخدم بتاً واحداً لكل خانة؟

الإجابة 12 • الصفحة 85

عدّ النواتج قبل استخدام بايز

يغيّر الاحتمال الشرطي المجموعة التي تعدّها.

تغدو الكواشف وإعادة المحاولات وتقديرات الموثوقية أوضح حين تتحول الاحتمالات إلى أعداد حالات.

تعني $P(A|B)$ نسبة أفراد مجموعة B الذين ينتمون أيضاً إلى A. وهي ليست عادة $P(B|A)$. فالمقام هو المجموعة المعلوم انطباقها مسبقاً.

في محاولات مستقلة باحتمال فشل p ، يكون احتمال عدم وقوع أي فشل في n محاولة هو $(1-p)^n$. اطرحه من الواحد لتحصل على احتمال وقوع فشل واحد على الأقل.

الاستقلال افتراض في النموذج. وكثيراً ما تقع الأعطال معاً حين تشترك في شبكة أو مصدر طاقة أو خدمة منقلة؛ لذلك قد يضلنا ضرب احتمالات مفترضة الاستقلال.

100,000 تشغيل

99

إنذارات صحيحة

999

إنذارات كاذبة

9.02%

من الإنذارات تشير لأعطال

تجيب الحساسية والمعدّل الأساسي عن سؤالين مختلفين.

احسبها خطوة بخطوة

1. افترض وجود أعطال في 100 عملية من أصل 100,000 تشغيل. يكتشف الكاشف 99 منها.
2. من بين 99,900 عملية سليمة، ينتج معدل إنذار كاذب 1% عدداً قدره 999 إنذاراً كاذباً.
3. يصبح مجموع الإنذارات 1,098. ولا يقابل عطلاً منها إلا $9.02\% = 99/1,098$. للمعدّل الأساسي أهميته.

$$P(\text{fault} | \text{alarm}) = \text{عدد الإنذارات الصحيحة} \div \text{عدد جميع الإنذارات}$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_13.c

```
#include <math.h>
#include <stdio.h>
double any_failure(double p, unsigned n) {
    if (!isfinite(p) || p < 0 || p > 1) return NAN;
    if (n == 0 || p == 0) return 0;
    if (p == 1) return 1;
    return -expm1((double)n * log1p(-p));
}
int main(void) {
    printf("%.6f\n", any_failure(0.001, 100));
    printf("alarm means fault: %.2f%%\n", 100.0 * 99 / 1098);
    return 0;
}
```

الناتج المتوقع

```
0.095208
alarm means fault: 9.02%
```

تُحافظ `log1p` و `expm1` على دقة مفيدة للاحتتمالات الصغيرة. يفترض هذا المثال بلغة C الدعم المعتاد لـ NaN في المنصة؛ وقد تفضل واجهة إنتاجية نتيجة حالة صريحة.

انتبه إلى الحدود

معدل اكتشاف 99% لا يعني أن 99% من التنبيهات صحيحة. ولا تحصر احتمالاً مستحيلاً بصمت في $[0,1]$ ؛ فهذا يخفي مدخلاً سيئاً.

جرب قبل قراءة الإجابة

تفشل كل من محاولتين مستقلتين باحتمال 0.1. ما احتمال فشلهما معاً؟ وما احتمال فشل واحدة منهما على الأقل؟

الإجابة 13 • الصفحة 86

احتفظ بصورة توزيع البيانات

لا يستطيع متوسط واحد وصف تجربة كل مستخدم.

تحتاج بيانات زمن الاستجابة والمستشعرات إلى ملخصات تناسب السؤال، خصوصاً حين تكون بعض المشاهدات متطرفة.

المتوسط الحسابي هو المجموع مقسوماً على العدد. ويصف الوسيط وسط المشاهدات المرتبة. ولا يكون أي منهما تلقائياً الملخص الأنسب لكل قرار.

يقيس التباين التشتت المربع. ويقسم تباين المجتمع على n عند وصف المجموعة المسردة كاملة؛ أما المقدّر المعتاد غير المتحيز لتباين العينة فيقسم على $n-1$ ، عند أخذ عينات مستقلة متطابقة التوزيع ذات تباين منه.

يصف نموذج برنولي ناتجاً واحداً بنعم أو لا. ويعدّ التوزيع ثنائي الحدين النجاحات في محاولات مستقلة متساوية الاحتمال. ويصف بواسون الأعداد في عملية مناسبة ذات معدل ثابت وزيادات مستقلة. وليس المنحنى الطبيعي قانوناً عاماً لزمن الاستجابة.

زمن الاستجابة المرصود بالمللي ثانية



احسبها خطوة بخطوة

1. للقيم $[1, 1, 1, 2, 2, 3, 4, 20, 40, 100]$ ms، المجموع 174 والمتوسط 17.4 ms.

2. القيمتان الوسطيتان 2 و3، ولذلك الوسيط 2.5 ms.

3. بالاستيفاء الخطي عند الموضع $q(n-1)$ ، تكون p_{90} مساوية لـ 46 ms. وقد تعطي اصطلاحات أخرى للمئينات أجوبة مختلفة لهذه العينة الصغيرة جداً.

$$\text{mean} = \text{sum} / n; \text{ sample variance} = \text{squared deviations} / (n-1)$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_14.cpp

```
#include <algorithm>
#include <cmath>
#include <iostream>
#include <stdexcept>
#include <vector>

double quantile(std::vector<double> x, double q) {
    if (x.empty() || !std::isfinite(q) || q < 0 || q > 1)
        throw std::domain_error("invalid quantile input");
    for (double v : x) if (!std::isfinite(v))
        throw std::domain_error("nonfinite sample");
    std::sort(x.begin(), x.end());
    const double p = q * static_cast<double>(x.size() - 1);
    const auto i = static_cast<std::size_t>(p);
    const auto j = std::min(i + 1, x.size() - 1);
    return std::lerp(x[i], x[j], p - static_cast<double>(i));
}

int main() {
    std::cout << quantile({1,1,1,2,2,3,4,20,40,100}, .9) << '\n';
}
```

الناتج المتوقع

46

يستهدف هذا التنفيذ التعليمي مجموعات بيانات اعتيادية في الذاكرة، يمكن تمثيل حجمها بدقة في double. ويضيف المشروع A المتوسط والوسيط وتشتت العينة.

انتبه إلى الحدود

لا تأخذ متوسط قيم p99 من خوادم مختلفة للحصول على p99 شامل. ادمج العينات الأصلية أو ملخصات توزيع متوافقة.

جرب قبل قراءة الإجابة

كلفة الإصابة ns 4 باحتمال 0.95، وكلفة الإخفاق ns 80 باحتمال 0.05. ما الكلفة المتوقعة؟ وهل تصف ذيل التوزيع؟

الإجابة 14 • الصفحة 86

قِسْ عدم اليقين، لا السرعة وحدها

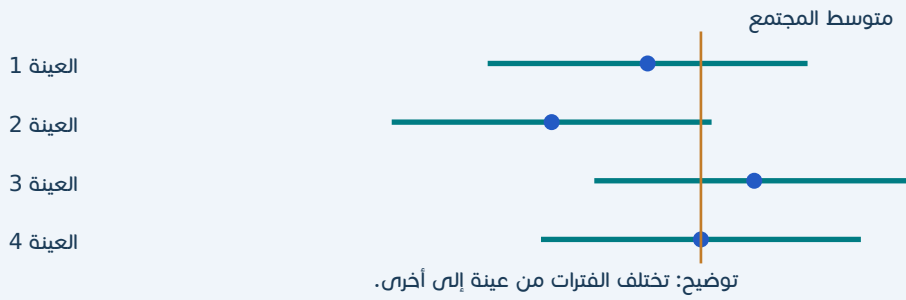
التسارع المقاس تقدير ناتج عن تجربة مشوشة.

قبل الاحتفاء بتحسين صغير، اسأل إن كانت التجربة تستطيع تمييزه من التباين الطبيعي.

تعطي المشاهدات المستقلة ذات التباين المنتهي خطأً معيارياً للمتوسط يُقدَّر بـ $s/\sqrt{n}$. ولا تفيد المشاهدات الإضافية إلا حين تضيف معلومات ذات معنى.

للمشاهدات الطبيعية المستقلة مجهولة التباين، تستخدم فترة ثقة ثنائية الطرف لمتوسطها معاملاً من توزيع ستودنت t . وللعينات الكبيرة، يعد 1.96 معاملاً تقريبياً شائعاً لمستوى 95% في شروط مناسبة.

تصف الثقة تكرار تغطية الطريقة للقيمة الحقيقية. ولا تعني وقوع 95% من الطلبات الفردية داخل الفترة. وقد تجعل عينات قياس الأداء المترابطة فترةً ساذجةً أضيق مما ينبغي.



احسبها خطوة بخطوة

1. افترض $n=100$ قياس مستقل، متوسطها 50 ms وانحرافها المعياري للعينة 10 ms.
2. الخطأ المعياري المقدَّر $10/\sqrt{100} = 1$ ms.
3. باستخدام t نحو 1.984 عند 99 درجة حرية، نحصل تقريباً على $[48.016, 51.984]$ ms للمتوسط، وفق النموذج المعلن.

$$SE = s / \sqrt{n}; \text{ interval} = \text{mean} \pm t * SE$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_15.c

```
#include <math.h>
#include <stdio.h>
int main(void) {
    const double mean_ms = 50.0;
    const double sample_sd_ms = 10.0;
    const double n = 100.0;
    const double t95 = 1.984; /* approximate, df = 99 */
    const double margin = t95 * sample_sd_ms / sqrt(n);
    printf("mean interval: [%.3f, %.3f] ms\n",
        mean_ms - margin, mean_ms + margin);
    return 0;
}
```

الناتج المتوقع

```
mean interval: [48.016, 51.984] ms
```

يحسب هذا المثال فترةً من إحصاءات ملخصة معطاة؛ وليس إطار قياس أداء، ولا تنفيذاً لدالة كميات توزيع t . ويجب أن يتغير المعامل مع مستوى الثقة وحجم العينة.

انتبه إلى الحدود

استخدم نتائج قابلة للرصد، وعملاً كافياً لكل عينة مؤقتة، وساعة رتيبة. عشو ترتيب النسخ أو قارنها في أزواج، وسجل خيارات المترجم والعتاد، واحتفظ بالتشغيلات البطيئة ما لم توجد قاعدة مبررة لاستبعادها.

جرب قبل قراءة الإجابة

كم مشاهدة مستقلة تحتاج إلى تنصيف الخطأ المعياري إذا ظل التباين الأصلي ثابتاً؟

الإجابة 15 • الصفحة 86

بادل الذاكرة بمعدل خطأ معلوم

تجري بنى البيانات الاحتمالية تقريباً مقصوداً.

تفيد مرشحات بلوم والخوارزميات العشوائية حين تكون افتراضاتها وكلفة أخطائها صريحتين.

تربط دالة التجزئة فضاء مفاتيح كبيراً بمخرجات أقل، ولذلك لا مفر من التصادمات. وفي ظل تجزئات مستقلة شبه منتظمة، ينمو احتمال التصادم تقريباً مع عدد الأزواج، لا مع عدد المفاتيح وحده.

يُضبط مرشح بلوم k موضع بت لكل مفتاح يُدرج في m بت. ويثبت بت مطلوب غير مضبوط عدم الوجود؛ أما ضبطها جميعاً فيعني احتمال الوجود. ويضمن الإدراج والبحث الصحيحان بتجزئات متسقة عدم وجود سلبيات كاذبة في التصميم الأساسي الذي لا يحذف.

لعدد n من المفاتيح المدرجة، التقريب الشائع هو $k = (m/\ln(2)) \ln(n)$. وتكون المفاضلة كفؤة قرب $k = m/\ln(2)$. يفترض النموذج مواضع فحص موزعة توزيعاً مناسباً.

بت مطلوب غير مضبوط يثبت عدم الوجود.



ضبط كل البتات المطلوبة يعني: افحص المخزن الأصلي.

احسبها خطوة بخطوة

1. خصص عشرة ملايين بت لمليون مفتاح: عشرة بتات لكل مفتاح، أي 1.25 MB عشرية.
2. عدد مواضع الفحص الأمثل تقريباً هو $\ln(2) = 0.693$ ؛ لذا جرب $k=7$.
3. معدل الإيجابيات الكاذبة التقريبي 0.00819، أي 0.819%. أكد الانتماء في المخزن المرجعي المعتمد.

$$p \sim (1 - \exp(-kn/m))^k$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_16.cpp

```
#include <cmath>
#include <iomanip>
#include <iostream>
int main() {
    const double bits_per_key = 10.0;
    const int probes = 7;
    const double occupied = -std::expm1(-probes / bits_per_key);
    const double fp = std::pow(occupied, probes);
    std::cout << std::fixed << std::setprecision(3)
               << 100.0 * fp << "%\n";
}
```

الناتج المتوقع

0.819%

هذا نموذج لتحديد الحجم، وليس تنفيذاً لمرشح. وتتبع العدد الفعلي للمفاتيح مهم: فإدراج مفاتيح إضافية يملأ بتات أكثر ويزيد الإيجابيات الكاذبة.

انتبه إلى الحدود

في المحاكاة، استخدم مولدًا وتوزيعًا مناسبين بدل `rng()%N` إن كان يسبب انحياز التردد. ولا يلزم تطابق تسلسل خرج توزيعات C++ عبر تطبيقات المكتبة، حتى مع البذرة نفسها للمولد.

جرب قبل قراءة الإجابة

إذا كُلفت كل إيجابية كاذبة عملية بحث في قاعدة بيانات، فكم عملية بحث إضافية تحدث تقريباً بين 100,000 استعلام عن مفاتيح غير موجودة؟

الإجابة 16 • الصفحة 86

المتجه قائمة ذات معنى هندسي

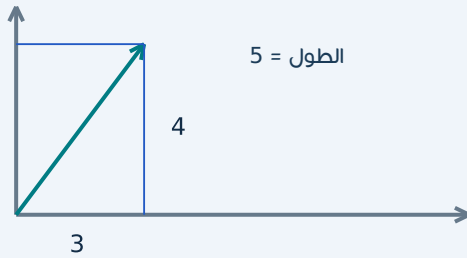
يقيس الضرب النقطي مدى توافق اتجاهين.

تدعم العملية نفسها الإضاءة، والبحث بالتشابه، والإسقاطات، وطبقات تعلّم الآلة.

المتجه قائمة مرتبة من المركبات. وللمتجه الهندسي أيضاً إطار إحداثي ووحدات. ويحتاج متجه السمات إلى ترتيب وتحجيم متسقين لسماته.

يضرب الضرب النقطي المركبات المتناظرة ويجمعها. وصيغته الهندسية $\text{length}(a) \times \text{length}(b) \times \cos(\text{angle})$. وللمتجهات الوحيدة يصبح تشابه جيب التمام.

المعيار الإقليدي هو الجذر التربيعي لمجموع مربعات المركبات. ولمرتبتين أو ثلاث، تساعد hypot على تجنب فيض النتائج الوسيطة واضمحلالها غير الضروريين. وتطبيع المتجه الصفري غير معرّف.



$$(3,4) / 5 = (0.6,0.8)$$

يحافظ متجه الوحدة على الاتجاه.

احسبها خطوة بخطوة

1. للمتجه $a=(3,4)$ طول 5. وقسمة كل مركبة على 5 تعطي $(0.6,0.8)$.
2. ضرب متجه الوحدة هذا نقطياً في $(1,0)$ يعطي 0.6، أي جيب تمام الزاوية بينهما.
3. يبقى إسقاط a على محور x المركبتين $(3,0)$. وتكون المركبة المحذوفة عمودية على هذا المحور.

$$a \cdot b = \sum(a[i]b[i]); \text{ cosine} = (a \cdot b) / (|a||b|)$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_17.c

```
#include <math.h>
#include <stdio.h>
int main(void) {
    const double x = 3.0, y = 4.0;
    const double length = hypot(x, y);
    if (!(length > 0) || !isfinite(length)) return 1;
    const double ux = x / length, uy = y / length;
    printf("length: %.1f\n", length);
    printf("unit: (%.1f, %.1f)\n", ux, uy);
    printf("cosine with x-axis: %.1f\n", ux);
    return 0;
}
```

الناتج المتوقع

```
length: 5.0
unit: (0.6, 0.8)
cosine with x-axis: 0.6
```

لهذا المثال مدخلات منتهية ومحدودة. وتحتاج المتجهات عالية الأبعاد إلى معيار محسوب مع التحجيم أو دالة جبر خطي موثوقة عندما تكون المقادير متطرفة.

انتبه إلى الحدود

لا يمكن مقارنة متجهين من إطارين إحداثيين مختلفين مقارنة ذات معنى حتى يتفق إطاراهما. وفي السمات، قد يهيمن مقياس وحدة كبير على الضرب النقطي.

جرب قبل قراءة الإجابة

احسب الضرب النقطي لـ $(1, 2, 3)$ و $(4, 0, -1)$. وما جيب تمام الزاوية بين $(1, 0)$ و $(0, 1)$ ؟

الإجابة 17 • الصفحة 86

افحص أبعاد المصفوفات قبل القيم

كل ضرب مصفوفات مجموعة من عمليات الضرب النقطي.

يكشف فحص الأبعاد أخطاء قبل فحص فهرس مصفوفة أو قيمة عددية.

يمكن لمصفوفة من m صفاً و n عموداً أن تتحول متجهاً ذا n مركبة إلى متجه ذي m مركبة. وخرج الصف i هو الضرب النقطي لذلك الصف في متجه المدخل.

يعطي ضرب A ذات الأبعاد $m \times k$ في B ذات الأبعاد $k \times n$ مصفوفة بأبعاد $m \times n$. ويتطابق البعدان الداخليان لأن كل ضرب نقطي يضم k حدًا.

يبحث حل $Ax=b$ عن متجه مجهول. لا تكون المعكوس صراحة لمجرد حل نظام. استخدم تحليلاً مناسباً: LU مع اختيار المحور للأنظمة المربعة العامة، وQR لكثير من مسائل المربعات الصغرى، وSVD حين تحتاج الرتبة أو حساسية المسألة إلى اهتمام خاص.

$$\begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix}_{2 \times 2} \times \begin{bmatrix} 4 \\ 5 \end{bmatrix}_{2 \times 1} = \begin{bmatrix} 13 \\ 19 \end{bmatrix}_{2 \times 1}$$

احسبها خطوة بخطوة

1. لتكن صفوف A هي $[2,1]$ و $[1,3]$ ، وليكن $x=[4,5]$.
2. الخرج الأول $13 = 2 \times 4 + 1 \times 5$ والثاني $19 = 1 \times 4 + 3 \times 5$.
3. وبالعكس، يستعيد حل $Ax=[13,19]$ المتجه $x=[4,5]$ هنا. وقد تجعل الصفوف شبه المعتمدة خطياً الاستعادة شديدة الحساسية لخطأ المدخلات.

$$A(m \times k) B(k \times n) \rightarrow C(m \times n)$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_18.cpp

```
#include <array>
#include <iostream>
int main() {
    const std::array<std::array<double, 2>, 2> a{{
        {2.0, 1.0}, {1.0, 3.0}
    }};
    const std::array<double, 2> x{4.0, 5.0};
    std::array<double, 2> y{};
    for (std::size_t i = 0; i < 2; ++i)
        for (std::size_t j = 0; j < 2; ++j)
            y[i] += a[i][j] * x[j];
    std::cout << y[0] << ' ' << y[1] << '\n';
}
```

الناتج المتوقع

13 19

تهيئة y بالصفر ضرورية قبل التجميع. وتجعل الأبعاد الثابتة كل حد حلقة وكل بُعد ظاهراً.

انتبه إلى الحدود

التخزين بترتيب الصفوف والجبر بمتجهات الصف اختيارات مختلفة. ولا يحدد ترتيب التخزين الجانب الذي يُضرب فيه المتجه. اكتب الاصطلاح الرياضي صراحة.

جرب قبل قراءة الإجابة

ما أبعاد حاصل ضرب دفعة X من 32 صفًا و8 سمات في أوزان W من 8 صفوف و3 مخرجات؟

الإجابة 18 • الصفحة 86

أبق الإحداثيات ضمن أطرها

لا يكون للموضع معنى إلا بالنسبة إلى إطار إحداثي.

تنشأ أخطاء الرسومات والروبوتات كثيراً من تركيب تحويلات صحيحة بترتيب خاطئ.

يحوّل دوران ثنائي الأبعاد بزاوية t النقطة (x, y) إلى $(\cos(t)x - \sin(t)y, \sin(t)x + \cos(t)y)$. وتستخدم الدوال المثلثية في المكتبة الراديان.

تنقل الإزاحة نقطة، لكنها لا تنقل متجه اتجاه. وتمثل الإحداثيات المتجانسة هذا الفرق بمركبة أخيرة تساوي 1 للنقاط و0 للاتجاهات.

مع متجهات الأعمدة، تعمل المصفوفة الأبعد يميناً أولاً. ويسمّي تحويل مثل $T_world_from_local$ المصدر والوجهة، مما يسهّل التحقق من التركيب.

دور، ثم أرح

$(1,0) \longrightarrow (2,1)$

أرح، ثم دور

$(1,0) \longrightarrow (0,3)$

قد تنتج العمليتان نفسيهما موضعين مختلفين.

احسبها خطوة بخطوة

1. دور النقطة $(1,0)$ بمقدار 90 درجة عكس عقارب الساعة. فتصبح $(0,1)$ ، مع مراعاة تقريب الفاصلة العائمة.
2. ثم أرحها بمقدار $(2,0)$. تكون النتيجة $(2,1)$.
3. اعكس الترتيب: أرحها إلى $(3,0)$ ثم دورها. تكون النتيجة $(0,3)$. فالترتيب جزء من العملية.

```
world_point = translation(rotation(local_point))
```

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_19.cpp

```
#include <cmath>
#include <iostream>
#include <numbers>
struct P { double x, y; };
P rotate(P p, double radians) {
    const double c = std::cos(radians), s = std::sin(radians);
    return {c*p.x - s*p.y, s*p.x + c*p.y};
}
int main() {
    P p = rotate({1, 0}, std::numbers::pi / 2);
    p.x += 2;
    std::cout << p.x << ' ' << p.y << '\n';
}
```

الناتج المتوقع

2 1

جيب تمام $\pi/2$ الرياضي يساوي صفرًا بدقة، لكن نتيجة الفاصلة العائمة قد تكون باقياً ضئيلاً. قارن باستخدام السماح الهندسية للتطبيق.

انتبه إلى الحدود

يجب أن تتضمن الواجهة اصطلاحاً يمينية الإطار أو يساريته، ووحدات الزوايا، واصطلاحات الصفوف والأعمدة. ويمكن للكواتيرنيون تمثيل دوران ثلاثي الأبعاد بإيجاز، لكنه يحتاج أيضاً إلى اصطلاح وسياسة للتطبيق.

جرب قبل قراءة الإجابة

لماذا لا يكون لإضافة إزاحة إلى متجه سرعة معنى في العادة؟ وما الوحدة المناسبة للسرعة الزاوية؟

الإجابة 19 • الصفحة 87

المشتقة حساسية محلية

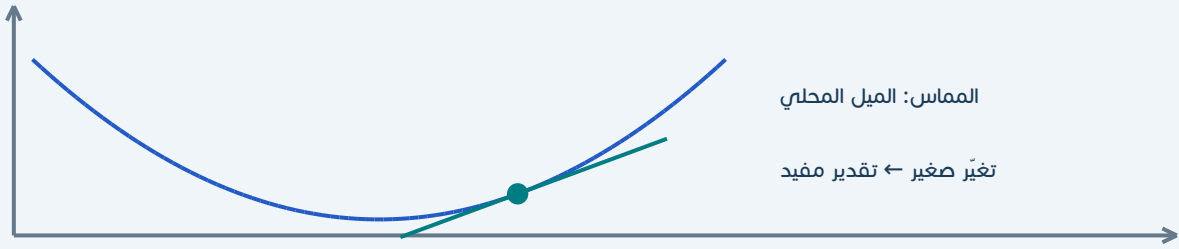
تخبرك المشتقة كيف يستجيب الخرج لتغير صغير في المدخل.

توجّه التدرجات عملية التحسين، وتكشف أيضاً متى تكون النتيجة حساسة لقياسات غير مؤكدة.

للدالة $y=f(x)$ ، المشتقة هي حد ميل التغيرات المتجاورة. وحول x يُنتج تغير صغير dx تغيراً في الخرج يساوي تقريباً $f'(x)dx$.

مع عدة مدخلات، يجمع التدرج مشتقة جزئية لكل مدخل. وتُسأل كل مركبة عما يحدث حين يتغير ذلك المدخل وتظل بقية المدخلات ثابتة.

تربط قاعدة السلسلة المراحل: إذا اعتمد y على u ، واعتمد u على x ، فإن $dy/dx = (dy/du)(du/dx)$. ويطبق الانتشار العكسي هذه القاعدة مراراً مع إعادة استخدام القيم الوسيطة.



احسبها خطوة بخطوة

- خذ $f(x)=(x-3)^2$. مشتقتها $2(x-3)$.
- عند $x=5$ يساوي الميل 4. ويتنبأ تغير قدره $+0.01$ بزيادة في الخرج تبلغ نحو 0.04.
- الزيادة الدقيقة $0.0401 - 4 = 4.0401 - 4$. والفرق الصغير هو حد الانحناء.

التغير المحلي $\approx$ الميل $\times$ تغير المدخل

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_20.c

```
#include <math.h>
#include <stdio.h>
double f(double x) { return (x - 3) * (x - 3); }
int main(void) {
    const double x = 5.0, h = 1e-5;
    const double numeric = (f(x+h) - f(x-h)) / (2*h);
    const double analytic = 2*(x-3);
    printf("analytic: %.6f\n", analytic);
    printf("numeric: %.6f\n", numeric);
    return 0;
}
```

الناتج المتوقع

```
analytic: 4.000000
numeric: 4.000000
```

يفيد الفرق المركزي في فحص مشتقة عند نقاط ملساء. وقد تزيد قيمة h الصغيرة للغاية خطأ التقريب العددي؛ قارن عدة أحجام معقولة للخطوة.

انتبه إلى الحدود

المشتقة ليست متوسط ميل على أي مجال اعتباطي. وعند نقطة زاوية غير قابلة للاشتقاق، قد لا يطابق فحص الفرق المركزي المشتقة التحتية المختارة للتدريب.

جرب قبل قراءة الإجابة

إذا كانت $y = \sin(2x)$ ، فما dy/dx ؟ وإذا كان لقياس نصف القطر خطأ صغير dr ، فما حساسية مساحة الدائرة πr^2 ؟

الإجابة 20 • الصفحة 87

درب نموذجاً بتقليل الخسارة

يضبط التعلم المعلمات لتقليل مقياس خطأ معن.

يكشف نموذج انحدار صغير الرياضيات المشتركة بين أنظمة تعلم أكبر بكثير.

للخط $\hat{y} = w \cdot x + b$ ، يحسب متوسط مربعات الخطأ متوسط مربعات البواقي. والباقي هو التنبؤ مطروحاً منه الهدف. وتخبّرنا الخسارة بما تحاول عملية التحسين تحسينه.

التدرجان هما متوسط $2 \cdot \text{residual} \cdot x$ بالنسبة إلى w ، ومتوسط $2 \cdot \text{residual}$ بالنسبة إلى b . احسبهما من المعلمات الحالية نفسها قبل تحديثهما.

يطرح النزول التدرجي حاصل ضرب معدل التعلم في التدرج. وقد يتذبذب المعدل الكبير جداً أو يتباعد؛ ويهدر المعدل الصغير جداً الدورات. ويغير حجم السمت هذه الهندسة.



احسبها خطوة بخطوة

1. للخسارة الأبسط $L(w) = (w-3)^2$ ، ابدأ من $w=0$ وبمعدل 0.1.

2. التدرج -6، فيصبح w مساوياً لـ $0.6 = 0 - 0.1 \cdot (-6)$.

3. يساوي كل خطأ جديد 0.8 من الخطأ السابق. وتستمر الخسارة في الانخفاض نحو الصفر لهذا المعدل وهذه الدالة التربيعية تحديداً.

```
parameter <- parameter - learning_rate x gradient
```

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_21.cpp

```
#include <iomanip>
#include <iostream>
int main() {
    double w = 0;
    const double rate = 0.1;
    for (int step = 0; step < 100; ++step) {
        const double gradient = 2 * (w - 3);
        w -= rate * gradient;
    }
    std::cout << std::fixed << std::setprecision(6) << w << '\n';
}
```

الناتج المتوقع

3.000000

تُقرَّب النتيجة إلى ست منازل عشرية؛ ولا تكون دقيقة رياضياً بعد عدد منتهٍ من الخطوات. ويوسِّع المشروع C هذه الحلقة إلى ملاءمة خط مع فحص مستقل للتدرج.

انتبه إلى الحدود

لا تكفي خسارة التدريب وحدها لإثبات القدرة على التعميم. افصل بيانات التحقق، واختر خط أساس، وافحص ما إذا كانت السمات ستتاح فعلاً عند التنبؤ.

جرب قبل قراءة الإجابة

لهذه الدالة التربيعية نفسها، ماذا يحدث عند معدل 0.5، ومعدل 1، ومعدل أكبر من 1؟

الإجابة 21 • الصفحة 87

يحوّل التراكم المعدلات إلى مجاميع

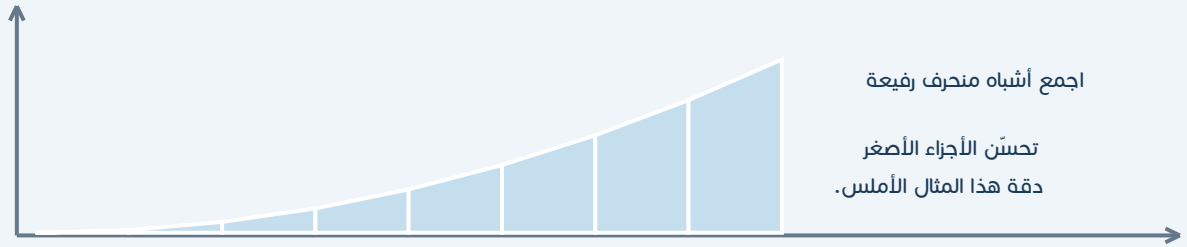
يجمع التكامل مساهمات صغيرة كثيرة.

تنشأ المساحة والطاقة والمحاكاة من جمع معدل على فترة زمنية أو مجال مكاني.

إذا كان المعدل ثابتاً، فالمجموع يساوي المعدل مضروباً في الفترة. وإذا تغيّر، فقسّم الفترة إلى أجزاء صغيرة، وقرب مساهمة كل جزء.

تستخدم قاعدة شبه المنحرف متوسط ارتفاعي طرفي كل جزء. وهي تكامل الخط المستقيم تكاملاً دقيقاً في الحساب الدقيق.

تصف المعادلة التفاضلية العادية معدل تغير حالة. وتستخدم طريقة أويلر الصريحة الحالة القديمة لكل مشتقة. أما تحديث السرعة قبل الموضع فيعطي طريقة مختلفة، تسمى كثيراً أويلر شبه الضمنية في الميكانيكا.



احسبها خطوة بخطوة

1. كامل $f(x)=x*x$ من 0 إلى 1 بجزأين متساويين عرض كل منهما 0.5.
2. مساحتا شبهي المنحرف هما $0.5*(0+0.25)/2$ و $0.5*(0.25+1)/2$.
3. المجموع 0.375؛ والتكامل الدقيق $1/3$. وتقلل الأجزاء الأدق الخطأ لهذه الدالة الملساء.

مساحة الجزء: $\text{width} \times (\text{left_height} + \text{right_height})/2$ تقريباً

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_22.c

```
#include <stdio.h>
double f(double x) { return x*x; }
int main(void) {
    const int n = 1000;
    const double h = 1.0 / n;
    double area = 0;
    for (int i = 0; i < n; ++i) {
        const double left = i*h, right = (i+1)*h;
        area += h * (f(left) + f(right)) / 2;
    }
    printf("%.9f\n", area);
    return 0;
}
```

الناتج المتوقع

0.333333500

الحدود وعدد الخطوات الموجب ثابتان هنا. وفي أداة تكامل قابلة لإعادة الاستخدام، تحقق من المجال، وافحص التقارب عند تصغير الخطوة.

انتبه إلى الحدود

لا تجعل الخطوة الزمنية الأصغر كل طريقة مستقرة تلقائياً لكل مسألة. فللمعادلة $x' = -k \cdot x$ ، لا تتناقص طريقة أويلر الصريحة إلا حين $0 < k \cdot dt < 2$.

جرب قبل قراءة الإجابة

للمعادلة $x' = -10x$ ، ما مجال الخطوات الزمنية الموجبة الذي يجعل أويلر الصريحة تتناقص؟ وماذا يحدث عند $dt=0.2$ ؟

الإجابة 22 • الصفحة 87

الإشارات جوارات مرجحة

يجمع المرشّح القيم المتجاورة بأوزان مختارة.

تتعمّ الفكرة نفسها الصوت، وتطمس الصور، وتظهر في الشبكات العصبية الالتفافية. يضرب مرشّح منتخ العينات في أوزان ويجمعها. وتحفظ الأوزان التي مجموعها واحد إشارة ثابتة بعيداً عن آثار سياسة الحدود.

يعكس الالتفاف الرياضي النواة أثناء تحريكها. وتنفذ واجهات كثيرة للصور والتعلّم الارتباط المتبادل دون عكس. وتعطي النوى المتناظرة النتيجة نفسها في الحالتين.

يفقد أخذ العينات معلومات إذا طُويت الترددات الأعلى إلى ترددات أدنى. وتحتاج الإشارة المحدودة النطاق إلى معدل عينات أكبر من ضعف أعلى تردد فيها لإعادة بنائها مثاليًا. ويحتاج أخذ العينات عملياً إلى مرشّح لمنع الطي وهامش انتقال.



احسبها خطوة بخطوة

1. استخدم الأوزان $[0.25, 0.5, 0.25]$ حول العينات $[0, 0, 4, 0, 0]$.
2. المخرجات الداخلية الثلاثة الصالحة هي $1, 2, 1$.
3. يحسب FFT تحويل فورييه المتقطع بكفاءة. ويعطي الضرب باستخدام FFT التفاضلاً دائرياً بطبيعته؛ ويلزم حشو كافٍ للحصول على التفاضل خطي.

$$y[i] = 0.25 \cdot x[i-1] + 0.5 \cdot x[i] + 0.25 \cdot x[i+1]$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_23.c

```
#include <stdio.h>
int main(void) {
    const double x[5] = {0, 0, 4, 0, 0};
    for (size_t i = 1; i < 4; ++i) {
        const double y = .25*x[i-1] + .5*x[i] + .25*x[i+1];
        printf("%.1f%s", y, i == 3 ? "\n" : " ");
    }
    return 0;
}
```

الناتج المتوقع

1.0 2.0 1.0

يصدر هذا المثال فقط المواضيع ذات الجوار الكامل. ويعرض المشروع B قاعدة حدود مستقلة وصريحة لصورة ثنائية الأبعاد.

انتبه إلى الحدود

قد لا يكون الطمس الصحيح لقيم الضوء الخطي صحيحاً لأعداد sRGB المرمّزة. فتفسير العينات جزء من الحساب.

جرب قبل قراءة الإجابة

في الالتفاف الخطي لسلسلتين طولاهما 4 و3، ما طول الخرج؟ وما مقدار الحشو الكافي لـ FFT؟

الإجابة 23 • الصفحة 87

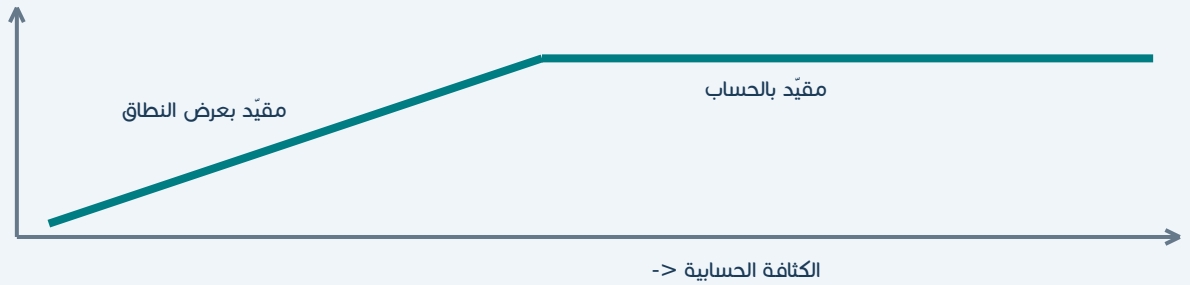
عدّ البايتات بعناية كعدّ العمليات

لا يستطيع معالج سريع الحساب ببيانات لم تصله.

كثيراً ما يشرح عرض النطاق ومحلية البيانات وإعادة الاستخدام الأداء أفضل من عدّ العمليات الحسابية وحدها. الكثافة الحسابية هي العمليات المفيدة لكل بايت يُنقل عبر حد ذاكرة مختار. سمّر هذا الحد: فحركة السجلات والذاكرة المخبئية والذاكرة الرئيسية أعداد مختلفة.

ينفذ جمع مصفوفتين من double وكتابة خرج واحد عملية جمع واحدة لكل 24 بايتاً مفيداً لكل عنصر. وقد تكون الحركة الفعلية أكبر بسبب سلوك الذاكرة المخبئية والتخصيص عند الكتابة.

سقف أداء بسيط هو الأصغر بين ذروة الحساب وعرض النطاق مضروباً في الكثافة الحسابية. هذا نموذج، وليس وعداً. وبحسّن تقسيم ضرب المصفوفات إلى كتل إعادة الاستخدام؛ وتزيد SIMD التوازي الحسابي، لكنها لا تخلق عرض نطاق.



احسبها خطوة بخطوة

1. يستوعب متجه من 256 بتاً أربع قيم double من 64 بتاً، ومتجه من 512 بتاً ثمانين قيم.
2. عند 50 مليار بايت مفيد/ثانية، يكون سقف عرض النطاق لتدفق يحتاج 24 بايتاً لكل جمع نحو 2.083 مليار عملية جمع/ثانية.
3. قد تخفض الوصلات غير المحاذاة والعناصر المتبقية وإخفاقات المخبأ واعتماديات الاختزال معدل الإنجاز المقاس أكثر.

$$\text{performance} \leq \min(\text{compute peak}, \text{bandwidth} \times \text{operations/byte})$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_24.c

```
#include <stdio.h>
int main(void) {
    const double bandwidth = 50e9;
    const double bytes_per_add = 3.0 * 8.0;
    const double ceiling = bandwidth / bytes_per_add;
    printf("ideal ceiling: %.3f Gadd/s\n", ceiling / 1e9);
    const size_t row = 7, col = 3, columns = 20;
    printf("row-major offset: %zu\n", row*columns + col);
    return 0;
}
```

الناتج المتوقع

```
ideal ceiling: 2.083 Gadd/s
row-major offset: 143
```

الإزاحة مقاسة بالعناصر، لا بالبايتات. وتحتاج الفهرسة الديناميكية إلى حدود الصفوف والأعمدة، فضلاً عن حجم تخصيص مفحوص. ولا يطابق تخطيط مصفوفة `std::vector` متصلة تخطيط متجهات متداخلة.

انتبه إلى الحدود

يتغير ترتيب اختزال الفاصلة العائمة مع التحويل إلى متجهات والتوازي. وقد تهمل خيارات `Fast-math` افتراضات تتطلبها المجاميع المعوّضة وفحوص القيم المنتهية.

جرب قبل قراءة الإجابة

ما الحد الأدنى المثالي لزمان مسح 5 GiB بسرعة 50 GiB/s؟ وهل تضمن ++C سطر ذاكرة مخبئية بحجم 64 بايتاً؟

الإجابة 24 • الصفحة 87

يبقى في العمل المتوازي جزء متسلسل

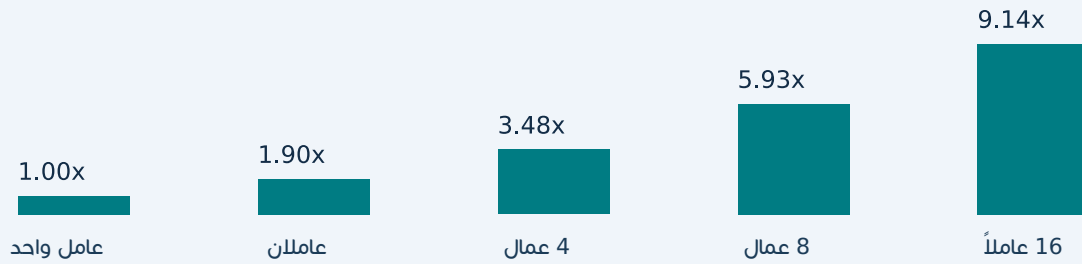
لا تفيد زيادة العمال إلا حيث يستطيع العمل أن يتداخل فعلاً.

تحتاج الخوادم والنوى المتوازية إلى نموذج لعبء العمل، وإلى مزامنة صحيحة أيضاً.

يقسم نموذج أمدال لعبء العمل الثابت التنفيذ إلى جزء متسلسل نسبته s ، وجزء قابل للتوازي المثالي نسبته $1-s$. ومع N عاملاً يكون التسارع المثالي $1/(s+(1-s)/N)$.

يربط قانون ليتل متوسط عدد العناصر على المدى الطويل في نظام مستقر بمتوسط معدل وصولها وزمن بقائها فيه: $L = \lambda * W$. استخدم حدود نظام ووحدات متسقة.

يتعلق التزامن الصحيح بالترتيب والعمر، لا بالحساب وحده. وسباق البيانات في $C++$ سلوكه غير معرّف. وقد تشارك متغيرات مستقلة سطر مخبأ وتزاحم عبر المشاركة الكاذبة.



احسبها خطوة بخطوة

1. إذا كان 5% من عبء ثابت متسلسلاً، فسقف التسارع مع عدد لا نهائي من العمال هو 20.
2. مع 16 عاملاً، التسارع المثالي $1/(0.05+0.95/16) = 9.143$.
3. عند 1,000 طلب/ثانية، ومتوسط بقاء 20 ms في النظام المقاس، يعطي قانون ليتل متوسط 20 طلباً موجوداً فيه.

$$S = 1/(s + (1-s)/N); L = \text{arrival_rate} \times \text{mean_time}$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_25.cpp

```
#include <iomanip>
#include <iostream>
int main() {
    const double serial = 0.05, workers = 16;
    const double speedup = 1 / (serial + (1-serial)/workers);
    const double arrival_rate = 1000, mean_seconds = 0.02;
    std::cout << std::fixed << std::setprecision(3)
        << "ideal speedup: " << speedup << '\n';
    std::cout << "mean in system: "
        << arrival_rate * mean_seconds << '\n';
}
```

الناتج المتوقع

```
ideal speedup: 9.143
mean in system: 20.000
```

يمثل هذا عبء عمل ثابت، ويتجاهل أعباء التوازي الإضافية. وقد يتصرف عبء عمل يتوسع مع الموارد بطريقة مختلفة. ويعطي قانون ليتل متوسطاً، لا سعة طابور ولا ضماناً لمئين.

انتبه إلى الحدود

لا تجعل المتغيرات الذرية الخوارزمية كلها صحيحة. يظل نشر البيانات وترتيب الذاكرة واستردادها والثوابت المركبة محتاجاً إلى تبرير مكتمل.

جرب قبل قراءة الإجابة

إذا كان $s = 0.10$ و $N = 8$ ، فما التسارع المثالي؟ ولماذا لا يكفي متوسط استغلال أقل من 100% لإثبات وفاء جدول زمني أنني بكل مواعيده النهائية؟

الإجابة 25 • الصفحة 88

جَهْز البيانات دون تسريب الإجابة

يجب ألا يتعلّم التحويل إلا من بيانات متاحة عند التدريب.

تبدأ النمذجة الجيدة قبل أول تدرج: بالوحدات، وتقسيم البيانات، والقيم المفقودة، وعقد معالجة مسبقة قابل لإعادة الإنتاج.

يطرح التقييس متوسط التدريب ويقسم على مقياس التدريب. وقد يمنع ذلك سمة كبيرة الوحدات من الهيمنة على المسافات والتدرجات.

قدّر معلمات المعالجة المسبقة على قسم التدريب وحده. وطبّق المعلمات المخزنة نفسها على بيانات التحقق والاختبار والمدخلات المستقبلية. فالتقسيم بعد ملائمة جميع البيانات يسرّب المعلومات.

تحتاج السمة ذات التباين الصغري إلى سياسة، مثل إنتاج صفر وتسجيل أنها ثابتة. ويكون التقسيم العشوائي خاطئاً كثيراً للبيانات المرتبة زمنياً، أو للمشاهدات المتكررة للكيان نفسه.



احسبها خطوة بخطوة

1. متوسط قيم التدريب [10,20,30] هو 20.
2. استخدام انحراف معياري سكاني لهذه القيم التدريبية الثلاث يعطي $\text{sqrt}((100+0+100)/3) = 8.164966$.
3. تصبح قيمة مستقبلية 40 مساوية لـ $(40-20)/8.164966 = 2.449490$. ويجوز لها الوقوع خارج نطاق التدريب.

$$z = (x - \text{training_mean}) / \text{training_scale}$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_26.cpp

```
#include <cmath>
#include <iomanip>
#include <iostream>
int main() {
    const double train[] = {10, 20, 30};
    double mean = 0;
    for (double x : train) mean += x / 3.0;
    double variance = 0;
    for (double x : train) variance += (x-mean)*(x-mean)/3.0;
    const double scale = std::sqrt(variance);
    if (!(scale > 0)) return 1;
    const double future = 40;
    std::cout << std::fixed << std::setprecision(6)
              << (future-mean)/scale << '\n';
}
```

الناتج المتوقع

2.449490

اصطلاح المقياس هنا هو الانحراف المعياري السكاني الوصفي لمجموعة التدريب. والبيانات صغيرة جداً ومحدودة؛ أما التدفقات الكبيرة فتحتاج إلى تنفيذ مستقر يعالج القيم تبعاً أو يمرّ على البيانات مرتين.

انتبه إلى الحدود

قد تجتمع دقة اختبار عالية مع تسريب للمعلومات، أو اختلال في توازن الفئات يجعلها غير مفيدة. قارن بخط أساس بسيط، واختر مقاييس تعكس القرار الفعلي.

جرب قبل قراءة الإجابة

إذا قُيِّست سمات التدريب مستقلة عن سمات التحقق، باستخدام المتوسط الخاص بكل قسم، فلماذا يُعد ذلك مشكلة؟

الإجابة 26 • الصفحة 88

حوّل الدرجات إلى خسارة مستقرة

قد تختلف الصيغ المتكافئة اختلافاً كبيراً في سلوكها العددي.

يعتمد التصنيف على الدوال الأسية واللوغاريتمات، حيث تمنع صيغة جبرية جيدة فيضاً يمكن تجنبه.

تحوّل Softmax درجات منتهية إلى أوزان موجبة مجموعها واحد في الحساب الدقيق. ويُبقي طرح أكبر درجة قبل الأس كل قيمة أسية عند الواحد أو دونه.

للفئة المستهدفة t ، تكون الإنتروبيا المتقاطعة $-\log(p[t])$. احسبها من الدرجات المزاحة بالصيغة $\log(\sum(\exp(z[j]-m))) - (z[t]-m)$ ، بدل أخذ لوغاريتم احتمال قد يُقرب إلى الصفر.

في التصنيف الثنائي، تستخدم سيغمويد مستقرة $1/(1+\exp(-z))$ عندما يكون z غير سالب، و $\exp(z)/(1+\exp(z))$ خلاف ذلك. وتظل احتمالات النموذج محتاجة إلى فحوص المعايرة.

احتمالات مستقرة للقيم [1000,1001,1002]



احسبها خطوة بخطوة

1. تؤدي الدرجات الخام [1000,1001,1002] إلى فيض عند حساب $\exp$ مباشرة في `binary64`.
2. اطرح 1002 لتحصل على $[-2,-1,0]$. وقيمها الأسية تقريباً $[0.1353,0.3679,1]$.
3. طبعها إلى $[0.0900,0.2447,0.6652]$. ولل فئة الأخيرة تكون الخسارة نحو 0.407606.

$$p[i] = \exp(z[i]-m) / \sum(\exp(z[j]-m))$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_27.cpp

```
#include <algorithm>
#include <array>
#include <cmath>
#include <iomanip>
#include <iostream>
int main() {
    const std::array<double,3> z{1000, 1001, 1002};
    const double m = *std::max_element(z.begin(), z.end());
    std::array<double,3> e{};
    double sum = 0;
    for (std::size_t i=0; i<z.size(); ++i) {
        e[i] = std::exp(z[i]-m);
        sum += e[i];
    }
    std::cout << std::fixed << std::setprecision(6);
    for (double v : e) std::cout << v/sum << ' ';
    std::cout << "\nloss: " << std::log(sum)-(z[2]-m) << '\n';
}
```

الناتج المتوقع

```
0.090031 0.244728 0.665241
loss: 0.407606
```

لهذه المدخلات المنتهية المحدودة، تتجنب الإزاحة بالقيمة العظمى الفيض. ويجب أن تحدد الواجهة العامة سلوك المدخل الفارغ، ودرجات Logits غير المنتهية، والهدف خارج النطاق؛ وقد تظل الفروق المتطرفة عرضة للاضمحلال إلى الصفر.

انتبه إلى الحدود

لا تكون أوزان Softmax احتمالات موثوقة للواقع تلقائياً. فقد يخطئ النموذج بثقة عالية في مدخلات تختلف عن بيانات تدريبه.

جرب قبل قراءة الإجابة

ماذا يحدث للخروج إذا أُضيف 50 إلى كل درجة؟ وماذا يحدث إذا ضُربت جميع الدرجات في 10؟

الإجابة 27 • الصفحة 88

الانتشار العكسي إعادة استخدام لقاعدة السلسلة

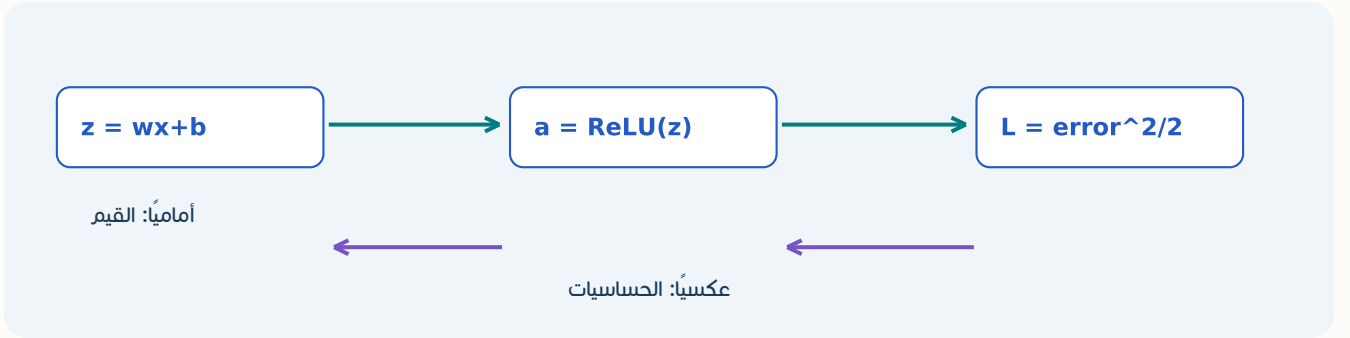
اتبع الحساب الأمامي، ثم أعد الحساسيات إلى الخلف.

يكفي عصبون واحد لفهم سبب نجاح الاشتقاق التلقائي.

لتكن $z = w \cdot x + b$ و $a = \max(0, z)$ و $L = (a - y)^2 / 2$. يجعل عامل النصف المشتقة الأولى للخسارة ببساطة $a - y$.

عندما $z > 0$ ، تكون مشتقة ReLU مساوية لـ 1. وعندما $z < 0$ تكون 0. وعند الصفر لا توجد المشتقة؛ فتختار تطبيقات التدريب اصطلاحاً، غالباً هو الصفر.

اضرب المشتقات المحلية: $dL/dw = (a - y) \cdot \text{ReLU}'(z) \cdot x$ و $dL/db = (a - y) \cdot \text{ReLU}'(z)$. ويشارك الاشتقاق بالنمط العكسي هذه الحساسيات الوسيطة بين معلمات كثيرة.



احسبها خطوة بخطوة

1. خذ $x=2$, $w=0.5$, $b=0$, $y=3$. يعطي المرور الأمامي $z=1$, $a=1$, $L=2$.

2. حساسية الخرج $a - y = -2$ وتمررها ReLU عند المدخل الموجب دون تغيير.

3. تدرج الوزن -4 وتدرج الانحياز -2. وعند معدل 0.1 تصبح القيمتان $w=0.9$ و $b=0.2$.

$$dL/dw = dL/da \times da/dz \times dz/dw$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_28.cpp

```
#include <algorithm>
#include <iostream>
int main() {
    const double x=2, target=3, rate=0.1;
    double w=0.5, b=0;
    const double z=w*x+b;
    const double a=std::max(0.0, z);
    const double dz=(a-target)*(z>0 ? 1.0 : 0.0);
    const double dw=dz*x, db=dz;
    w -= rate*dw;
    b -= rate*db;
    const double next=std::max(0.0, w*x+b);
    std::cout << "w: " << w << " b: " << b << '\n';
    std::cout << "new loss: " << .5*(next-target)*(next-target) << '\n';
}
```

الناتج المتوقع

```
w: 0.9 b: 0.2
new loss: 0.5
```

ينتج التدرجان من المرور الأمامي نفسه. والخرج الجديد 2 والخسارة 0.5. إن تحسن خطوة واحدة فحص محلي، وليس برهاناً على نجاح التدريب دائماً.

انتبه إلى الحدود

يخلط تحديث w قبل حساب تدرج يعتمد على w القديمة بين حالتين للمعاملات. أبق القيم الأمامية والتدرجات متسقة، خصوصاً عبر الدفعة.

جرب قبل قراءة الإجابة

إذا كان z سالباً تماماً في هذا المثال، فما التدرجات المارة عبر ReLU؟ ولماذا نتجنب فحص مشتقة عند $z=0$ بالضبط؟

الإجابة 28 • الصفحة 88

يشارك التشابه والانتباه في الضرب النقطي

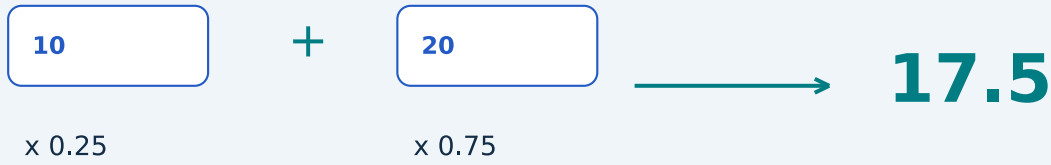
الانتباه متوسط مرجح يعتمد على البيانات.

تغدو التضمينات والانتباه ملموسين حين نفصل عمليتهما المتجهية عن النموذج الذي تعلّمهما.

التضمين (Embedding) تمثيل متجهي متعلّم أو مصمّم لغرض ما. ويقارن تشابه جيب التمام الاتجاهات بعد التطبيق؛ أما الضرب النقطي الخام فيعتمد على المقدار أيضاً.

يُكوّن انتباه الضرب النقطي المحجّم الدرجات $QK^T/\sqrt{d}$ ، ويطبق Softmax عبر المفاتيح لكل استعلام، ثم يمزج متجهات القيم V بهذه الأوزان.

مع مركبات مستقلة، متوسطها صفر وتباينها واحد، ينمو تباين الضرب النقطي مع البعد d . وتضبط القسمة على $\sqrt{d}$ هذا المقياس وفق هذا النموذج التفسيري. وليس ذلك ضماناً عاماً للتوزيع.



يبقى المتوسط المرجح المحدّب داخل نطاق القيم.

احسبها خطوة بخطوة

1. ينتج استعلام واحد الدرجتين $[0, \ln(3)]$ لمفتاحين. وتعطي الدالة الأسية $[1, 3]$.
2. أوزان Softmax هي $[0.25, 0.75]$. وإذا كانت القيم العددية المقابلة 10 و20، فالخرج 17.5.
3. مع كثرة الاستعلامات والمفاتيح، يُنشئ الانتباه الكثيف القياسي مصفوفة درجات من الاستعلامات $\times$ المفاتيح؛ وقد ينمو حجمها تربيعياً مع طول التسلسل.

$$\text{attention}(Q, K, V) = \text{softmax}(QK^T / \sqrt{d}) V$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_29.cpp

```
#include <cmath>
#include <iostream>
int main() {
    const double s0=0, s1=std::log(3.0);
    const double m=s1;
    const double e0=std::exp(s0-m), e1=std::exp(s1-m);
    const double w0=e0/(e0+e1), w1=e1/(e0+e1);
    const double output=w0*10+w1*20;
    std::cout << w0 << ' ' << w1 << '\n';
    std::cout << output << '\n';
}
```

الناتج المتوقع

```
0.25 0.75
17.5
```

يعزل هذا المثال خطوة الترجيح. ويقدم المشروع D مثال استرجاع صغيراً كاملاً بمتجهات مكتوبة يدوياً؛ ولا يدعي تدريب تضمينات دلالية.

انتبه إلى الحدود

يمنع القناع السببي الوصول إلى المواضيع المستقبلية. وإذا حُجبت كل المفاتيح، فلن يكون لـ Softmax المعتادة توزيع صالح؛ ويحتاج التنفيذ إلى سياسة صريحة.

جرب قبل قراءة الإجابة

إذا كانت أبعاد Q هي 5x4، وأبعاد K هي 7x4، وأبعاد V هي 7x3، فما أبعاد مصفوفتي الدرجات والخرج؟

الإجابة 29 • الصفحة 88

اجعل التكميم قراراً في ميزانية الخطأ

تشتري البتات الأقل الكفاءة بقبول تقريب مقاس.

يحتاج الاستدلال بالأعداد الصحيحة وصيغ البيانات المدمجة إلى قواعد صريحة للمقياس والقص والتقريب والمجمّع. يحوّل المكمّم التآلفي قيمة حقيقية إلى $q = \text{round}(x/s) + z$ ، ثم يقص q إلى نطاق عدد صحيح مخزن. وتعيد إزالة التكميم $s \cdot (q - z)$. ويجب أن يكون المقياس s موجباً.

دون قص، يعطي التقريب إلى الأقرب خطأ مطلقاً لا يتجاوز $s/2$ في الحساب الدقيق. وقد يسبب القص خطأ أكبر بكثير خارج المجال الممثل.

تتراكم حواصل ضرب الصحيحة عادة في نوع أوسع. ولمقادير متناظرة حدها 127، لا يتجاوز مقدار كل حاصل ضرب 16,129؛ ويحدد طول الضرب النقطي ما إذا كان المجموع يتسع. كما تهم حدود الانحياز ونقطة الصفر.

0.26 -> 0.30

أقرب نقطة شبكة: المقياس = 0.1



احسبها خطوة بخطوة

1. خذ المقياس $s=0.1$ ونقطة الصفر $z=0$ ، بأعداد صحيحة مخزنة من -127 إلى 127.
2. عند $x=0.26$ يعطي التقريب إلى الأقرب $q=3$. وإعادة البناء 0.3، فيكون الخطأ 0.04.
3. عند $x=20$ يعطي القص $q=127$ وإعادة بناء 12.7. والخطأ 7.3 خطأ قص، وهو أكبر بكثير من $s/2$.

$$\hat{x} = \text{scale} \times (\text{quantized} - \text{zero_point})$$

من الفكرة إلى الشفرة

الملف المرافق: examples/lesson_30.cpp

```
#include <algorithm>
#include <cmath>
#include <iostream>
int main() {
    const double x=0.26, scale=0.1;
    const double rounded=std::round(x/scale);
    const double bounded=std::clamp(rounded, -127.0, 127.0);
    const int q=static_cast<int>(bounded);
    const double restored=scale*q;
    std::cout << "q: " << q << " restored: " << restored << '\n';
    std::cout << "error: " << std::abs(restored-x) << '\n';
}
```

الناتج المتوقع

```
q: 3 restored: 0.3
error: 0.04
```

تجعل المدخلات المنتهية والمحدودة تحويل النوع آمناً. وتستخدم `std::round` تقريب حالات المنتصف بعيداً عن الصفر؛ وتستخدم صيغ استدلال كثيرة سياسة أخرى. طابق الصيغة المستهدفة بدل افتراض اتفاقها.

انتبه إلى الحدود

اختر بيانات معيارية تمثل مدخلات النشر، ثم قيس جودة المهمة النهائية إلى جانب الخطأ العددي. ولا تضمن الموترات الأصغر زمن استجابة أقل على كل محرّك تنفيذ.

جرب قبل قراءة الإجابة

مع مقياس 0.02 ودون قص، ما حد خطأ التقريب إلى الأقرب؟ وهل يبرر التخزين في `int8` تجميع ضرب نقطي طويل في `int8`؟

الإجابة 30 • الصفحة 88

تقرير صادق لزمان الاستجابة

اجمع الملخصات والمئينات والعناية العددية.

تتلقى أزمئة استجابة عشرة طلبات بالمللي ثانية. والمهمة الأولى وصف هذه العينة تحديداً، لا تقرير أداء كل طلب مستقبلي.

يحسب البرنامج المتوسط والانحراف المعياري للعينة في مرور واحد بتحديث ويلفورد (Welford)، ثم يرتب نسخة لحساب مئينات مستوفاة. ويظهر الذيل الكبير الفرق بين زمن الاستجابة المعتاد والكلفة الإجمالية.

تحتفظ طريقة ويلفورد بمتوسط جارٍ، وبالمجموع $M2$ لمربعات الانحرافات عن ذلك المتوسط المتغير. تغيّر كل مشاهدة جديدة المتوسط أولاً، ثم تضيف $\text{delta}*(x\text{-new_mean})$ إلى $M2$. وهذا يتجنب طرح مجموعين هائلين متقاربين جداً.

قرارات تسبق التنفيذ

عقد المدخلات. قائمة ثابتة غير فارغة من أزمئة استجابة منتهية وغير سالبة. المثال صغير؛ ويجب التحقق من البيانات الخارجية قبل تجميعها.

قاعدة المئينات. استخدم الموضع $q*(n-1)$ في القائمة المرتبة، واستوف بين القيم المتجاورة. صرّح بهذه القاعدة عند مقارنة الأدوات.

قاعدة التباين. استخدم $M2/(n-1)$ لتباين العينة. يتطلب ذلك عينتين على الأقل؛ فلا يمكن تقدير تشتت العينة من مشاهدة واحدة.

التفسير. صف المشاهدات. ولا تصطنع فترة ثقة من عشر قيم شديدة الالتواء دون نموذج مناسب.

روابط بين الأفكار

استخدم أدوات الدروس عن قصد: الوحدات والنطاقات أولاً، ومدخل محسوب يدوياً ثانياً، ثم برنامج كامل وفحوص ينبغي أن تظل صحيحة بعد التغييرات.

project_a_latency.cpp

البرنامج كاملاً

```
#include <algorithm>
#include <cmath>
#include <iomanip>
#include <iostream>
#include <vector>
int main() {
    std::vector<double> x{1,1,1,2,2,3,4,20,40,100};
    std::size_t n=0;
    double mean=0, m2=0;
    for (double v : x) {
        if (!std::isfinite(v) || v<0) return 1;
        ++n;
        const double delta=v-mean;
        mean += delta/static_cast<double>(n);
        m2 += delta*(v-mean);
    }
    if (n<2) return 1;
    std::sort(x.begin(), x.end());
    const auto quantile = [&](double q) {
        const double p=q*static_cast<double>(n-1);
        const auto i=static_cast<std::size_t>(p);
        const auto j=std::min(i+1, n-1);
        return std::lerp(x[i], x[j], p-static_cast<double>(i));
    };
    std::cout << std::fixed << std::setprecision(3);
    std::cout << "n: " << n << '\n';
    std::cout << "mean: " << mean << " ms\n";
    std::cout << "sample sd: "
        << std::sqrt(m2/static_cast<double>(n-1)) << " ms\n";
    std::cout << "median: " << quantile(.5) << " ms\n";
    std::cout << "p90: " << quantile(.9) << " ms\n";
    std::cout << "max: " << x.back() << " ms\n";
}
```

ابن هذا الملف وشغله بوصفه برنامجاً مستقلاً. ويوفر مشروع CMake المصاحب هدفاً تنفيذياً منفصلاً بالاسم الأساسي نفسه.

اشرح معنى الخرج

الناتج المتوقع

```
n: 10
mean: 17.400 ms
sample sd: 31.638 ms
median: 2.500 ms
p90: 46.000 ms
max: 100.000 ms
```

المتوسط 17.4 ms، رغم أن سبع قيم من العشر لا تتجاوز 4 ms. وتسهم مشاهدة 100 ms بأكثر من نصف الزمن الإجمالي.

يصف الوسيط الطلب الأوسط بصورة أفضل هنا. وتحدد القيمة العظمى أسوأ طلب مرصود؛ لكنها ليست حداً لزمان الاستجابة المستقبلي.

نتيجة p90 اصطلاح استيفاء، وليست بالضرورة مشاهدة وقعت فعلاً. ومع عشر عينات فقط، تكون تقديرات الذيل شديدة الخشونة.

فحوص ذات معنى

1. استبدل القائمة بعشر قيم متطابقة مقدار كل منها 5 ms: يجب أن يكون المتوسط وكل مئين 5، والانحراف المعياري صفراً.

2. أضف 10 إلى كل عينة: ينتقل المتوسط والمئينات بمقدار 10، بينما يظل الانحراف المعياري ثابتاً في حدود التقريب.

3. اضرب جميع المشاهدات في اثنين: يجب أن تتضاعف كل الملخصات المعلنة ذات الوحدات الزمنية.

تقدّم خطوة إضافية

حمل بعد ذلك قياسات من اختبار أداء خاص بك. وأبقِ الوحدات الأصلية والعدد والبيئة وطريقة الجمع مع التقرير. وللتدفقات الضخمة، استخدم ملخصاً تقريبياً موثقاً للكميات (Quantile Sketch) وسجل عقد خطئه.

المشروع B

مرشح صورة صغير بلغة C الخالصة

حوّل متوسطاً مرجحاً محلياً إلى تحويل مرئي.

يمكن التعامل مع الصورة بوصفها حقلاً ثنائي الأبعاد من الأعداد. يطمس هذا المشروع صورة رمادية اصطناعية بأبعاد 5×5 باستخدام مرشح صندوق 3×3 .

يحتوي المدخل على قيمة مركزية مضيئة واحدة مقدارها 9. ويحسب كل خرج متوسط الجيران المتاحين. وعند حدود الصورة، يستخدم هذا التنفيذ الجيران الصالحين فقط، ويقسم على عددهم الفعلي.

يكتب البرنامج في مصفوفة خرج مستقلة. أما تحديث المدخل في مكانه فيتيح للبكسلات اللاحقة قراءة قيم سبق طمسها، فيغيّر العملية الرياضية.

قرارات تسبق التنفيذ

معنى المدخلات. قيم رمادية صغيرة ذات شدة خطية. وهي ليست بايتات SRGB مرمّزة، ولذلك لا نحتاج إلى تحويل دالة النقل.

النواة. أوزان متساوية عبر جوار 3×3 . في الداخل يساوي كل وزن $1/9$.

قاعدة الحدود. اقتطع الجوار عند الحد وأعد التطبيع. تختلف هذه القاعدة مكانياً قرب الحدود، ولا تطابق الحشو بالأصفر.

التخزين. تجعل مصفوفتان ثابتتان 5×5 الحدود صريحة. وتحتاج الصور الديناميكية إلى أبعاد وأحجام تخصيص مفحوصة.

روابط بين الأفكار

استخدم أدوات الدروس عن قصد: الوحدات والنطاقات أولاً، ومدخل محسوب يدوياً ثانياً، ثم برنامج كامل وفحوص ينبغي أن تظل صحيحة بعد التغييرات.

project_b_blur.c

البرنامج كاملاً

```

#include <stdio.h>
int main(void) {
    enum { H=5, W=5 };
    const double input[H][W] = {
        {0,0,0,0,0}, {0,0,0,0,0}, {0,0,9,0,0},
        {0,0,0,0,0}, {0,0,0,0,0}
    };
    double output[H][W] = {{0}};
    for (int r=0; r<H; ++r) {
        for (int c=0; c<W; ++c) {
            double sum=0;
            int count=0;
            for (int dr=-1; dr<=1; ++dr) {
                for (int dc=-1; dc<=1; ++dc) {
                    const int rr=r+dr, cc=c+dc;
                    if (rr>=0 && rr<H && cc>=0 && cc<W) {
                        sum += input[rr][cc];
                        ++count;
                    }
                }
            }
            output[r][c]=sum/count;
        }
    }
    for (int r=0; r<H; ++r) {
        for (int c=0; c<W; ++c)
            printf("%.1f%s", output[r][c], c==W-1 ? "\n" : " ");
    }
    return 0;
}

```

ابن هذا الملف وشغله بوصفه برنامجاً مستقلاً. ويوفر مشروع CMake المصاحب هدفاً تنفيذياً منفصلاً بالاسم الأساسي نفسه.

اشرح معنى الخرج

الناتج المتوقع

```
0.0 0.0 0.0 0.0 0.0
0.0 1.0 1.0 1.0 0.0
0.0 1.0 1.0 1.0 0.0
0.0 1.0 1.0 1.0 0.0
0.0 0.0 0.0 0.0 0.0
```

تتوزع القيمة المركزية على المواضع الداخلية التسعة التي تحتوي جواراتها عليها. ويصبح كل منها 1 في هذا المثال.

يبقى المدخل الثابت ثابتاً، بما في ذلك عند الحدود، لأن كل خرج يقسم على عدد جيرانه الفعلي.

يُحفظ السطوع الإجمالي في هذه النبضة المركزية تحديداً. لكن قاعدة الحدود هذه لا تحفظ المجموع الكلي لكل صورة ممكنة؛ ويكشف نقل النبضة قرب حافة هذا الفرق.

فحوص ذات معنى

1. استبدل كل بكسل بالقيمة 7: يجب أن يساوي كل خرج 7.
2. انقل النبضة إلى زاوية، واحسب المخرجات الأربع غير الصفية يدوياً: $9/4$ و $9/6$ و $9/6$ و $9/9$.
3. تحقق من أن تغيير مدخل واحد لا يؤثر إلا في مواضع الخرج التي تضم جواراتها ذلك المدخل.

تقدّم خطوة إضافية

قارن بعد ذلك الحشو بالأصفر، والانعكاس، وتثبيت القيم عند الحدود. هذه تعريفات رياضية مختلفة. ولصورة إنتاجية، عرّف فضاء الألوان وتخطيط القنوات والتقريب قبل التحسين.

المشروع C

لائم خطأ وافحص تدرجه

ابن حلقة تعلّم كاملة وحتمية.

أزواج التدريب هي $x = [-2, -1, 0, 1, 2]$ و $y = [-3, -1, 1, 3, 5]$ وهي تتبع $y = 2x + 1$. نعرف الإجابة كي نستطيع تدقيق عملية التعلّم.

يتنبأ النموذج بـ $w \cdot x + b$. وخسارته متوسط مربعات البواقي. نحسب التدرجات من كل نقطة تدريب، ونأخذ متوسطها، ثم نحدّث المعلمتين معاً.

قبل التدريب، يفحص فرق منتهٍ مركزي مشتقة الوزن عند $w = 0, b = 0$. وينبغي أن تتفق مع التدرج التحليلي، الذي يساوي نحو 8-. يكشف هذا الاختبار عوامل مفقودة وأخطاء الإشارة.

قرارات تسبق التنفيذ

تصميم البيانات. تسهّل المدخلات المتمركزة حول الصفر فحص الحساب. وتحتاج البيانات الحقيقية إلى تقسيم مقصود، ومعالجة مسبقة تخص التدريب وحده.

الخسارة. متوسط مربعات الخطأ دون عامل النصف. لذلك تحتوي المشتقتان كلاًهما على عامل 2.

معدل التعلّم. تناسب القيمة 0.1 هذه الدالة التربيعية المحدودة. وليست قيمة افتراضية قابلة للنقل إلى سمات عامة.

معياري النجاح. تقترب المعلمات من $w = 2$ و $b = 1$ ، وتتنبأ نقطة مستبعدة من التدريب $x = 3$ بالقيمة 7. تعرض هذه النتائج الاصطناعية آلية العمل، لا جودة نموذج في الواقع.

روابط بين الأفكار

استخدم أدوات الدروس عن قصد: الوحدات والنطاقات أولاً، ومدخل محسوب يدوياً ثانياً، ثم برنامج كامل وفحوص ينبغي أن تظل صحيحة بعد التغييرات.

project_c_regression.cpp

البرنامج كاملاً

```

#include <array>
#include <iomanip>
#include <iostream>
const std::array<double,5> xs{-2,-1,0,1,2};
const std::array<double,5> ys{-3,-1,1,3,5};
double loss(double w, double b) {
    double total=0;
    for (std::size_t i=0; i<xs.size(); ++i) {
        const double r=w*xs[i]+b-ys[i];
        total += r*r;
    }
    return total/static_cast<double>(xs.size());
}
int main() {
    const double h=1e-5;
    const double numeric=(loss(h,0)-loss(-h,0))/(2*h);
    double w=0, b=0;
    for (int step=0; step<200; ++step) {
        double dw=0, db=0;
        for (std::size_t i=0; i<xs.size(); ++i) {
            const double r=w*xs[i]+b-ys[i];
            dw += 2*r*xs[i];
            db += 2*r;
        }
        const double n=static_cast<double>(xs.size());
        w -= 0.1*dw/n;
        b -= 0.1*db/n;
    }
    std::cout << std::fixed << std::setprecision(6);
    std::cout << "numeric dw at start: " << numeric << '\n';
    std::cout << "w: " << w << " b: " << b << '\n';
    std::cout << "training loss: " << loss(w,b) << '\n';
    std::cout << "prediction at 3: " << 3*w+b << '\n';
}

```

ابن هذا الملف وشغله بوصفه برنامجاً مستقلاً. ويوفر مشروع CMake المصاحب هدفاً تنفيذياً منفصلاً بالاسم الأساسي نفسه.

اشرح معنى الخرج

الناتج المتوقع

```
numeric dw at start: -8.000000
w: 2.000000 b: 1.000000
training loss: 0.000000
prediction at 3: 7.000000
```

عند المعلمات الابتدائية، تكون البواقي y -. ويعطي جمع $2 \times \text{residual} \times x$ والقسمة على خمسة القيمة -8 ؛ ويؤكد الفرق المنتهي ذلك محلياً.

يبدأ تدرج الانحياز عند -2 . وتتحرك المعلمتان نحو الخط المعلوم. والخسارة الصفرية المطبوعة مقربة، وليست ادعاءً بحساب دقيق.

إذا ضُربت المدخلات في ألف دون ضبط المعدل، فقد يصبح تدرج الوزن أكبر بكثير. يغيّر مقياس السمات سلوك التحسين.

فحوص ذات معنى

1. افحص المشتقتين عند نقطة ثانية مثل $w=7, b=-2$ ، لا عند الصفر فقط.
2. أضف ثابتاً c إلى كل هدف: ينبغي أن يزداد الانحياز الملائم بمقدار c ، بينما يظل الميل قريباً من 2 .
3. قارن التنبؤات بخط أساس ذي متوسط ثابت، وقيّم على بيانات لم تُستخدم في الملاءمة.

تقدّم خطوة إضافية

أضف بعد ذلك ضوؤاً مضبوطة، وافصل مجموعة اختبار. وسجّل خسارتي التدريب والاختبار منفصلتين. ولمسائل مربعات صغرى أكبر أو سيئة التكييف، استخدم QR أو SVD بدل حساب معكوس X -transpose- X صراحة.

استرجع بالاتجاه، ثم اشرح الأوزان

افحص الحساب وراء بحث متجهي مصغّر.

لثلاثة عناصر متجهات توضيحية ثنائية الأبعاد. والاستعلام هو $[1,1]$. نطبع الاستعلام وكل عنصر، ونحسب درجات جيب التمام، ثم نرتب العناصر.

المتجهات بيانات تعليمية مكتوبة يدوياً. ولا يولد البرنامج تضمينات دلالية، ولا يفهم اللغة، ولا يتحقق من نموذج استرجاع حقيقي.

نكوّن أيضاً أوزان Softmax مستقرة عند درجة حرارة 0.5. ويستخدم الترتيب درجات جيب التمام؛ وتوضح الأوزان كيف تمزج الدرجات نفسها قيماً مرتبطة بها.

قرارات تسبق التنفيذ

المتجهات. $A=[1,0]$, $B=[1,1]$, $C=[0,1]$. يطابق اتجاه B الاستعلام تماماً؛ ويتعادل A و C.

التطبيع. ارفض المتجهات صفرية الطول. كل المركبات هنا صغيرة ومنتهية.

الترتيب. جيب تمام أعلى يعني نتيجة أفضل. وترتب الدرجات المتساوية بمعرّف العنصر لعرض حتمي.

معنى الأوزان. هذه الأوزان المطبّعة ليست احتمالات صحة العنصر. وتتحكم درجة الحرارة في التركيز، ويجب أن تكون موجبة.

روابط بين الأفكار

استخدم أدوات الدروس عن قصد: الوحدات والنطاقات أولاً، ومدخل محسوب يدوياً ثانياً، ثم برنامج كامل وفحوص ينبغي أن تظل صحيحة بعد التغييرات.

project_d_retrieval.cpp

البرنامج كاملاً

```
#include <algorithm>
#include <array>
#include <cmath>
#include <iomanip>
#include <iostream>
struct Item { char id; double x, y, score, weight; };
int main() {
    std::array<Item,3> items{{
        {'A',1,0,0,0}, {'B',1,1,0,0}, {'C',0,1,0,0}
    }};
    const double qx=1/std::sqrt(2.0), qy=qx;
    const double temperature=0.5;
    double max_score=-1;
    for (auto& item : items) {
        const double norm=std::hypot(item.x,item.y);
        if (!(norm>0) || !std::isfinite(norm)) return 1;
        item.score=qx*(item.x/norm)+qy*(item.y/norm);
        max_score=std::max(max_score,item.score);
    }
    double sum=0;
    for (auto& item : items) {
        item.weight=std::exp((item.score-max_score)/temperature);
        sum += item.weight;
    }
    for (auto& item : items) item.weight /= sum;
    std::sort(items.begin(),items.end(),[](const Item& a,const Item& b){
        if (a.score!=b.score) return a.score>b.score;
        return a.id<b.id;
    });
    std::cout << std::fixed << std::setprecision(6);
    for (const auto& item : items)
        std::cout << item.id << " cosine=" << item.score
            << " weight=" << item.weight << '\n';
}
```

ابن هذا الملف وشغله بوصفه برنامجاً مستقلاً. ويوفر مشروع CMake المصاحب هدفاً تنفيذياً منفصلاً بالاسم الأساسي نفسه.

اشرح معنى الخرج

الناتج المتوقع

```
B cosine=1.000000 weight=0.473186
A cosine=0.707107 weight=0.263407
C cosine=0.707107 weight=0.263407
```

يمثل B الاتجاه الأقرب. ويُحسم تعادل A و C بقاعدة المعرّف الصريحة، لا بترتيب عرضي للحاوية. يغيّر تعديل درجة الحرارة الأوزان، لكنه لا يغيّر الترتيب ما دامت موجبة. ولا يغيّر ضرب متجه عنصر في ثابت موجب جيب التمام.

مع كثرة المتجهات المخزنة، تكون كلفة المسح الدقيق $O(\text{number_of_items} \times \text{dimension})$. وتفاضل الفهارس التقريبية بين معدل استرجاع النتائج الصحيحة (Recall) والذاكرة وزمن الاستجابة؛ اختبر هذه المفاضلة باستعلامات ممثلة.

فحوص ذات معنى

1. استخدم الاستعلام $[1,0]$: ينبغي أن يأتي A أولاً.
2. اضرب B في عشرة: ينبغي أن تظل درجات جيب التمام ثابتة في حدود التقريب.
3. تحقق من أن كل وزن غير سالب، وأن المجموع قريب من الواحد. وارفض الاستعلام الصفري قبل التطبيع.

تقدّم خطوة إضافية

استبدل بعد ذلك المتجهات التعليمية بتضمينات حقيقية من نموذج محدد، وقيّمها مقابل أحكام الصلة. وأبق إصدار النموذج والتطبيع ومقياس المسافة متسقة بين العناصر المخزنة والاستعلامات.

تابع التعلّم / دليل تخصصي

الرسومات والهندسة

احتفظ بالمعنى الفيزيائي وراء الإحداثيات.

امزج الضوء في الفضاء الصحيح

لا تعني قيمة sRGB قدرها 0.5 نصف الضوء الخطي للقيمة 1. فك ترميز دالة النقل، واحسب الضوء في الفضاء الخطي، ثم أعد الترميز للعرض. ولمزج متساوٍ من الأسود والأبيض في الضوء الخطي، تكون نتيجة sRGB المرمّزة نحو 0.735، لا 0.5.

استخدم اختبار الاتجاه بعناية

في بُعدين، تحدد إشارة $(bx-ax)(cy-ay)-(by-ay)(cx-ax)$ الجانب الذي تقع فيه نقطة بالنسبة إلى خط موجّه. وقرب الاستقامة المشتركة، قد يغيّر تقرب الفاصلة العائمة هذه الإشارة. تحتاج الاختبارات الهندسية المتينة إلى تحليل خطأ أو دقة تكيفية، لا إلى إبسيلون اعتباطية تُلصق بكل مقارنة.

الدورانات ليست متوسطات عادية

تمثل الكواتيرنيونات الوحيدة الدورانات ثلاثية الأبعاد بإيجاز. ويمثل q و $-q$ الدوران نفسه. ويجب أن يراعي الاستيفاء هذا التكافؤ ويحافظ على التطبيق. أما مجرد أخذ متوسط المركبات الأربع فقد يختار مساراً غير مقصود، أو ينتج متجهاً صفرياً.

حساب صغير مفيد

لمتجه عمودي وحدي على سطح $n=(0,1,0)$ واتجاه ضوء وحدي $l=(0,0.6,0.8)$ ، تكون $\max(0, n \cdot l)$ $=0.6$. وإذا أغفلت التطبيق، فستقيس القيمة طول المتجه أيضاً، مما يغيّر نموذج الإضاءة المقصود.

استخدم النموذج عن قصد

سؤال تحقق: حدّد الوحدات والإطار والتطبيق وفضاء الألوان قبل السؤال عن أسرع معادلة.

تابع التعلّم / دليل تخصصي

التحكم والمستشعرات والروبوتات

عدم اليقين والتوقعات جزء من الحالة.

ادمج القياسات بنموذج

يمكن دمج قياسين مستقلين غير متحيزين للكمية العددية نفسها، بتباينين موجبين معلومين v_1 و v_2 ، باستخدام أوزان معكوس التباين. للقيمتين 10 و 14 بتباينين 1 و 4، تكون القيمة المدمجة $10.8 = (10/1 + 14/4)/(1 + 1/4)$ ، وتباينها $0.8 = 1/(1 + 1/4)$.

يغيّر الارتباط الإجابة

المستشعرات التي تتشارك انحياز معايرة ليست مستقلة. وقد يجعل التعامل معها بوصفها مستقلة عدم اليقين المقدّر صغيراً بصورة غير واقعية. وتصف مصفوفات التباين كيف تتحرك الأخطاء معاً؛ وللعناصر خارج القطر أهميتها.

اجعل فترة أخذ العينات ظاهرة

يستخدم متحكم PID متقطع $\text{integral} + \text{error} * dt = \text{derivative} / dt$ وتغيّر dt الصفرية أو غير المنتظمة الحساب. ويتطلب تشبع المشغل استراتيجية لمنع تراكم التكامل (Anti-windup)، ويحتاج حد المشتقة كثيراً إلى ترشيح الضوضاء.

اطرح سؤال الاستقرار

يتناقص تحديث خطي متقطع $x_{\text{next}} = a * x$ عندما $|a| < 1$. وفي أبعاد متعددة، يتعلق الشرط المناظر للنظام الخطي بالقيم الذاتية لمصفوفة التحديث. ولا يكون متحكم يعمل في محاكاة بطيئة آمناً تلقائياً على العتاد.

استخدم النموذج عن قصد

سؤال تحقق: حول كل قياس إلى إطار مشترك، وسجل عدم يقينه، واختبر حدود أخذ العينات والمشغلات الفعلية.

الترجمات والبروتوكولات والمعلومات

تحوّل البنية المتقطعة الأنظمة الكبيرة إلى قواعد صريحة.

اعثر على نقطة ثابتة

يستخدم تحليل المتغيرات الحية $live_in = use \cup (live_out \text{ minus } def)$. وتكوّن مجموعات $live_in$ للعقد اللاحقة مجموعة $live_out$. ومع دالة انتقال رتيبة على مجموعة منتهية من الحقائق الممكنة، تتقارب قائمة عمل ذات جدولة صحيحة. ويجب أن تراعي النتيجة رسم تدفق التحكم وشروط الحدود.

ميّز كشف الخطأ من إثبات الهوية

يستخدم CRC حساب كثيرات الحدود على البتات، حيث يكون الجمع هو XOR. ويكشف فئات من أخطاء النقل العرضية وفق كثيرة الحدود والبروتوكول المحددين. ولا يوثّق المرسل ولا يقاوم التعديل المتعمد؛ إذ تحتاج المصادقة إلى بناء تشفير مناسب وتنفيذ خاضع للمراجعة.

اقرأ الإنترنت بوصفها متوسط المعلومات

لنتائج ممكنة باحتمالات p ، تكون إنتروبيا شانون $-\sum(p \cdot \log_2(p))$ ، مع اعتبار الحد ذي الاحتمال الصفري صفراً. وللبت العادل بت واحد من الإنترنت؛ وللنتائج المؤكد صفر. وليس هذا تلقائياً تقديراً لقوة كلمة مرور، ولا ضماناً لتعذر التنبؤ.

حدّد حجم خط نقل شبكي

لمسار 10 Gbit/s وزمن ذهاب وإياب 40 ms، يكون حاصل عرض النطاق والتأخير $10e9 \cdot 0.040/8 = 50,000,000$ بايت. وهو مقياس مفيد للحمولة قيد النقل. وقد تظل آليات التحكم في الازدحام، والفقد، وأعباء البروتوكول، وسعة المستقبل محدّدة لمعدل النقل.

استخدم النموذج عن قصد

سؤال تحقق: حدّد الرسم البياني، أو الحقل الحسابي، أو النموذج الاحتمالي، أو حدود الشبكة. فقد يمثل الرمز نفسه أنظمة مختلفة جداً.

المال والزمن والحساب الحتمي

تبدأ الدقة التامة بقاعدة من مجال التطبيق.

خزّن وحدة صريحة

تمثل السنوات الصحيحة مضاعفات جزء من مئة تمثيلاً دقيقاً ما بقي العدد الصحيح في النطاق. ولا تستخدم كل عملة أو كمية منزلتين عشريتين. ويظل الجمع بحاجة إلى فحص، وتحتاج حسابات الضريبة أو الفائدة إلى موضع تقريب وسياسة له.

اختر متى تقرّب

بندان بقيمة 5 سنتات لكل منهما، خاضعان لضريبة 10%، ينتجان ضريبة 0.5 سنت لكل بند. ومع تقريب المنتصف الموجب إلى الأعلى، يعطي تقريب كل بند ضريبة كلية قدرها سنتان؛ بينما يعطي الجمع أولاً ثم التقريب سنتاً واحداً. ولا يحدد أي التعبيرين سياسة العمل بذاته.

افصل مفاهيم الزمن

المدة فترة زمنية؛ والطابع الزمني المدني تسمية تقويمية لها تفسير مرتبط بمنطقة زمنية. استخدم ساعة رتيبة لقياس الزمن المنقضي. فقد تتغير ساعات الحائط بعد المزامنة، وقد تكون الأوقات التقويمية المحلية ملتبسة أو غير موجودة.

عامل الفاصلة الثابتة بوصفها صيغة تمثيل

للعديدين الخامين a و b بمقياس 2^F ، يمثل حاصل ضربهما $a*b$ بمقياس $2^{(2F)}$. اضرب في نوع واسع بما يكفي، وأعد التحجيم بقاعدة تقريب موثقة، وافحص نطاق الوجهة. وللحساب المشبع والحساب الملتف معنيان مختلفان.

استخدم النموذج عن قصد

سؤال تحقق: اكتب سياسات التقريب والفيض والمنطقة الزمنية بجوار تعريف النوع، كي لا يضطر قارئ لاحق إلى تخمينها بصورة خاطئة.

راجع استدلالك

قارن الافتراضات إلى جانب العدد النهائي. فقد تعطي سياسة حدود مختلفة إجابة أخرى صحيحة.

01 / امنح كل عدد معنى

$2.5165824 = 2,500,000,000 / 8 \times 1,048,576 \times 750$ ثانية. يجب تحويل البتات والبايتات قبل القسمة.

02 / حوّل المعادلة إلى دالة

$t = 15/60 = 0.25$ ، ولذلك $y = 1750$. وللقيمة 80، يعطي الاستقراء 4500؛ ويعطي عقد الحصر عند الحدود 4000؛ ويرفضها عقد التحقق.

03 / افهم النمو بالأسس واللوغاريتمات

تميّز 20 بتاً حتى 1,048,576 حالة. والسعة التالية $2^{13} = 8,192$.

04 / افهم البتات، ثم اكتب الشرط

$0xA7 = 1010\ 0111$ والشرط هو $(!cached \ \&\& \ !local) \ || \ !ready$.

05 / افحص النطاق قبل الضرب

يتطلب الجمع $a \leq \text{SIZE_MAX}$. وللقيمة $A > 0$ التي هي قوة لل اثنين، اشترط أولاً $x \leq \text{SIZE_MAX}$. ثم استخدم $\sim(A-1)$ و $(x+A-1)$.

06 / افهم الدورات ومقاييس التمثيل

القاسم المشترك الأكبر هو 21. ويعطي الطرح بترديد 10 نبضات. ودون قيد على الزمن المنقضي، لا يمكن التمييز بين 10 و 266 والقيم اللاحقة.

راجع استدلالك

قارن الافتراضات إلى جانب العدد النهائي. فقد تعطي سياسة حدود مختلفة إجابة أخرى صحيحة.

07 / الفاصلة العائمة مسطرة متحركة

لا. هذان العددان هما 2^{53} و $2^{53}+1$. والعدد الصحيح الأكبر التالي القابل للتمثيل بعد 2^{53} هو $2^{53}+2$.

08 / اعمل ضمن ميزانية خطأ صريحة

تجنب $\log_{1p}$ فقدان x عند تكوين $1+x$. ولا تستطيع دقة حسابية إضافية استعادة معلومات لم يحتوها القياس أصلاً.

09 / المجموع حلقة لها اسم

المجاميع هي 10 و 0 و 6. ولا يضم $(2,2)$ أي عنصر، ولذلك مجموعه صفر.

10 / أثبت صحة حلقة في جملة

تعيد 0 و 3 و 4. وثابت مسح القيمة العظمى هو: تساوي best القيمة العظمى للبائدة غير الفارغة التي عولجت بالفعل.

11 / توقّع العمل قبل قياس الزمن

دون ترتيب: $120 = (3 \times 2 \times 1) / 10 \times 9 \times 8$. ومع الترتيب: $720 = 10 \times 9 \times 8$. وقد تستبعد القيود بعض التوليفات.

12 / اكشف البنية بالمجموعات والرسوم البيانية

عادةً أنسب لهذا الرسم القليل الحواف. $125,000,000,000 = 10^{12} \text{ bits} / 8$ بايت، أي نحو 116.4 GiB قبل الأعباء الإضافية. وتكون قوائم التجاور

راجع استدلالك

قارن الافتراضات إلى جانب العدد النهائي. فقد تعطي سياسة حدود مختلفة إجابة أخرى صحيحة.

13 / عدّ النواتج قبل استخدام بايز

فشلهما معاً: $0.01 = 0.1 \times 0.1$. وفشل واحدة على الأقل: $1 - 0.9^2 = 0.19$. وقد تبطل الأسباب المشتركة كلا الحسابين المبنيين على استقلال المحاولات.

14 / احتفظ بصورة توزيع البيانات

القيمة المتوقعة $7.8 \text{ ns} = 0.05 \times 80 + 0.95 \times 4$. ولا تخبرنا وحدها بشيء عن سلاسل طويلة مترابطة أو مئين محدد في الذيل.

15 / قس عدم اليقين، لا السرعة وحدها

أربعة أمثال العدد، لأن الخطأ المعياري يتناسب مع $1/\sqrt{n}$. ولا يحقق تكرار مشاهدات مترابطة هذا التحسن تلقائياً.

16 / بادل الذاكرة بمعدل خطأ معلوم

نحو 819 وفق هذا النموذج. وقد يغيّر توزيع الاستعلامات وجودة التجزئة والتشيع العدد المرصود.

17 / المتجه قائمة ذات معنى هندسي

الضرب النقطي $1 = -3 + 0 + 4$. وجيب تمام الزاوية بين متجهي الوحدة المتعامدين صفر.

18 / افحص أبعاد المصفوفات قبل القيم

أبعاد XW هي 3×32 . ينتج كل صف ثلاثة مخرجات، ويستخدم كل خرج ضرباً نقطياً من ثمانية حدود.

راجع استدلالك

قارن الافتراضات إلى جانب العدد النهائي. فقد تعطي سياسة حدود مختلفة إجابة أخرى صحيحة.

19 / أبقِ الإحداثيات ضمن أطرها

تغيّر الإزاحة الأصل، بينما تصف السرعة تغير الموضع. وتقاس السرعة الزاوية بالراديان لكل وحدة زمن، وغالباً rad/s.

20 / المشتقة حساسية محلية

$dy/dx = 2 \cos(2x)$. وخطأ المساحة التقريبي $2\pi r dr$ ؛ ويكون الخطأ النسبي للمساحة نحو ضعف الخطأ النسبي لنصف القطر عندما r غير صفري.

21 / درّب نموذجاً بتقليل الخسارة

عند 0.5 تصل إلى $w=3$ في خطوة واحدة بحساب دقيق. وعند 1 تنقلب إشارة الخطأ دون أن يتقلص. وفوق 1 ينمو مقداره ما لم تكن عند الحل الأمثل بالفعل.

22 / يحوّل التراكم المعدلات إلى مجاميع

$0.2 < dt < 0$. وعند $dt=0.2$ يكون المعامل -1، فتتبادل الإشارة دون تناقص في الحساب الدقيق.

23 / الإشارات جوارات مرّجة

طول الخرج $6 = 4 + 3 - 1$. استخدم طول FFT لا يقل عن 6، وغالباً طولاً أكبر مدعوماً وملائماً.

24 / عدّ البايتات بعناية كعدّ العمليات

الحد الأدنى وفق عرض النطاق وحده 0.1 s. وحجم سطر المخبأ خاصية عتادية، وليس ضماناً عاماً من ++C.

راجع استدلالك

قارن الافتراضات إلى جانب العدد النهائي. فقد تعطي سياسة حدود مختلفة إجابة أخرى صحيحة.

25 / يبقى في العمل المتوازي جزء متسلسل

التسارع $4.706 = 1/(.1 + .9/8)$. وتعتمد المواعيد النهائية أيضاً على أنماط إطلاق المهام، والأولويات، والحجب، وتذبذب التوقيت، وأزمة التنفيذ في أسوأ حالة.

26 / جهّز البيانات دون تسريب الإجابة

يرى النموذج تحويلات مختلفة، ويكون التحقق قد استخدم معلومات غير متاحة لمدخل مستقبلي منفرد. خزّن معلمات التدريب وأعد استخدامها.

27 / حوّل الدرجات إلى خسارة مستقرة

لا تغيّر إضافة ثابت مشترك Softmax في الحساب الدقيق. ويغيّر الضرب التكرار؛ ومع قيمة عظمى وحيدة، يمنحها معامل موجب كبير وزناً أكبر.

28 / الانتشار العكسي إعادة استخدام لقاعدة السلسلة

التدرجات عبر ReLU غير النشطة صفر. وعند الصفر لا توجد مشتقة وحيدة، لذلك قد يختلف فحص الفرق المنتهي مع مشتقة تحتية مختارة.

29 / يشترك التشابه والانتباه في الضرب النقطي

أبعاد QK^T هي 5×7 . وبعد التطبيع صفّاً صفّاً، يعطي الضرب في V خرجاً بأبعاد 5×3 .

30 / اجعل التكميم قراراً في ميزانية الخطأ

الحد 0.01 في الحساب الدقيق. لا: حتى حاصل ضرب واحد قد يتجاوز int8، ويحتاج المجموع إلى حد مبني على الطول ونطاقات المدخلات.

أطلس المعادلات

الأعداد والخوارزميات

الوحدات

$$\text{amount} = \text{rate} \times \text{time}$$

استخدم وحدات متوافقة، وميّز البتات من البايتات.

التحويل التآلفي

$$t = (x - a) / (b - a); y = c + t(d - c)$$

اشترط $a \neq b$ ، وحدّد التحقق من المجال أو الحصر عند حدوده.

عدد الحالات

عدد N من الأعلام البوليانية يعطي 2^N حالة

قد تقلل القيود مجموعة الحالات الممكن الوصول إليها.

الضرب المفحوص

$$b \leq \text{MAX}/a \text{ أو } a = 0$$

للأعداد الصحيحة غير السالبة، قبل الضرب.

الجمع المفحوص

$$b \leq \text{MAX} - a$$

للقمتين غير الموقعتين a و b ، قبل الجمع.

استعلام المجاميع البادئة

$$\text{sum}[l, r] = P[r] - P[l]$$

اشترط $0 \leq l \leq r \leq n$ ؛ ويجب أن تتسع المجاميع.

الأزواج غير المرتبة

$$n(n-1)/2$$

اختزل قبل الضرب، وافحص النطاق.

القاسم المشترك الأكبر / المضاعف المشترك الأصغر

$$\text{lcm}(a, b) = (a / \text{gcd}(a, b)) * b$$

للأعداد الصحيحة الموجبة؛ ويجب أن يتسع حاصل الضرب.

المحاذاة إلى قوة لل اثنين

$$(x + A - 1) \& \sim(A - 1)$$

تكون $A > 0$ قوةً لل اثنين؛ ويجب أن يتسع الجمع.

عنصر بتخزين مرتب حسب الصفوف

$$\text{offset} = \text{row} * \text{columns} + \text{column}$$

تحقق من الفهارس، ومن سلامة حساب الحجم الإجمالي.

عدم اليقين والأداء

إعادة محاولات مستقلة

$$P(\text{any failure}) = 1 - (1-p)^n$$

يقع p في $[0,1]$ ، و n غير سالب؛ وللاستقلال أهميته.

الاحتمال الشرطي

$$P(A|B) = P(A \text{ and } B)/P(B)$$

اشتراط $P(B) > 0$ ؛ والمقام هو المجموعة المعلومة.

القيمة المتوقعة

$$E[X] = \text{sum}(\text{value} \times \text{probability})$$

لا تتطلب خطية التوقع استقلالاً.

تشئت العينة

$$s^2 = \text{sum}((x - \text{mean})^2)/(n-1)$$

اشتراط $n \geq 2$ ؛ وللتقييم المستقر أهميته.

عدم اليقين في المتوسط

$$SE \sim s/\sqrt{n}$$

مشاهدات مستقلة، وتباين منته.

نموذج بلوم

$$p \sim (1 - \exp(-kn/m))^k$$

عدد البتات m ، والمفاتيح n ، ومواضع الفحص k ؛ وهو نموذج للتجزئة.

تسارع عبء العمل الثابت

$$1/(s + (1-s)/N)$$

يمثل s الجزء المتسلسل؛ والعمل المتوازي مثالي في النموذج.

قانون ليتل

$$L = \text{arrival_rate} \times \text{mean_time}$$

متوسطات مستقرة على المدى الطويل، ضمن حدود النظام نفسها.

سقف الذاكرة

$$\text{operations/s} \leq \text{bytes/s} \times \text{operations/byte}$$

يجب التصريح بنموذج حركة البيانات ومستوى الذاكرة.

حاصل عرض النطاق والتأخير

$$\text{payload bytes} \sim \text{bit_rate} \times \text{RTT} / 8$$

استخدم الثواني والبتات/ثانية؛ ولا تشمل الصيغة الآثار الإضافية.

أطلس المعادلات

الهندسة والتعلم

الضرب النقطي

$$a \cdot b = \sum(a[i] \cdot b[i])$$

تطابق الأبعاد والأطر ومعاني السمات.

تشابه جيب التمام

$$(a \cdot b) / (|a| |b|)$$

متجهات غير صفرية؛ وطَّعَ بطريقة متينة.

أبعاد المصفوفات

$$(m \times k)(k \times n) \rightarrow (m \times n)$$

كل خرج ضرب نقطي من k حدًا.

خطوة التدرج

$$\theta \leftarrow \theta - \text{rate} \cdot \text{gradient}$$

تُحسب التدرجات من حالة واحدة متسقة للمعلمات.

الخسارة المربعة

$$L = \text{mean}((\text{prediction} - \text{target})^2)$$

حساسية للبواقي الكبيرة؛ وحدد الوحدات.

Softmax مستقرة

$$\exp(z[i] - \max(z)) / \sum(\exp(z - \max(z)))$$

مدخلات منتهية غير فارغة؛ وحدد الحالات المتطرفة.

خسارة التصنيف

$$\log(\sum(\exp(z - m))) - (z[\text{target}] - m)$$

تجنب أخذ لوغاريتم احتمال اضمحل إلى الصفر.

قاعدة السلسلة

$$dy/dx = (dy/du)(du/dx)$$

للتراكيب القابلة للاشتقاق.

أويلر الصريحة

$$x_{\text{next}} = x + dt \cdot f(t, x)$$

الحالة القديمة في جميع المشتقات؛ ويعتمد الاستقرار على dt .

التكثير التآلفي

$$q = \text{clip}(\text{round}(x/s) + z); \hat{x} = s(q - z)$$

اشتراط $s > 0$ ؛ وحدد النطاق والتقريب ونوع المجمِّع.

المسرد / الجزء الأول A-O

أبق المعنى قريباً

الخطأ المطلق (Absolute Error). المسافة عن مرجع، مقاسة بالوحدة نفسها.

الكلفة المطفأة (Amortized Cost). متوسط كلفة العملية عبر سلسلة عمليات، لها حد إجمالي حتمي.

الانحياز (Bias). إزاحة منهجية؛ ويعني أيضاً معلمة تضاف في النموذج.

حساسية المسألة (Conditioning). حساسية الإجابة الرياضية لتغيرات المدخلات.

فترة الثقة (Confidence Interval). إجراء يولّد فترة، وله تغطية معلنة عند تكرار أخذ العينات ضمن افتراضاته.

التغاير (Covariance). مقياس للتغير المشترك بين كميتين.

المجال (Domain). المدخلات المسموح بها لدالة.

التدرج (Gradient). متجه المشتقات الجزئية لدالة ذات خرج عددي واحد.

الثابت (Invariant). عبارة يُحافظ على صحتها عند نقطة محددة في الحساب.

درجة التصنيف الخام (Logit). درجة تصنيف غير مطبّعة قبل سيغمويد أو Softmax.

المعيار (Norm). مقياس لمقدار المتجه يحقق قواعد المعيار.

الفيضان (Overflow). نتيجة خارج النطاق القابل للتمثيل لصيغتها الحسابية.

أبق المعنى قريباً

المئين (Percentile). موضع في توزيع، يحتاج تعريفه في العينة إلى اصطلاح.

التكميم (Quantization). ربط القيم بمجموعة متقطعة قابلة للتمثيل.

الرتبة (Rank). عدد الصفوف أو الأعمدة المستقلة في مصفوفة.

الباقى (Residual). فرق، مثل التنبؤ مطروحاً منه المشاهدة.

العينة (Sample). قيم مرصودة مسحوبة أو مجمعة من عملية أو مجتمع.

الاستقرار (Stability). كيفية ضبط الخوارزمية للخطأ العددي؛ وفي الديناميكيات، كيفية تطور الاضطرابات.

الموتر (Tensor). في هذا الكتاب، مصفوفة متعددة الأبعاد ذات فهارس، تستخدمها البرمجيات العددية وبرمجيات التعلم.

الاضمحلال العددي (Underflow). دخول نتيجة فاصلة عائمة ضئيلة إلى نطاق الأعداد دون الطبيعية، أو تقريبها إلى الصفر، بحسب الصيغة والعملية.

الوحدة (Unit). مقياس كمية مقيسة ونوعها.

التباين (Variance). القيمة المتوقعة لمربع الانحراف عن المتوسط، أو تقدير محدد من العينة.

التحويل إلى متجهات (Vectorization). معالجة عناصر متعددة بعمليات متوازية على البيانات.

نقطة الصفر (Zero Point). الرمز الصحيح الذي يمثل الصفر الحقيقي في التكميم التآلفي.

تعمّق في القراءة لهدف

تدعم هذه المراجع الأولية قواعد اللغة والافتراضات الإحصائية ومعادلات الذكاء الاصطناعي في الدروس ذات الصلة. وهي طرق إلى دراسة أعمق، وليست متطلبات سابقة للكتاب.

R1 / مسودة لجنة C11 رقم N1570

WG14، 2011. الترقيات الصحيحة، والتعبيرات، وباقي القسم، والإزاحات، ومكتبة الرياضيات. تستخدم هذه الطبعة C11 خط أساس، ولا تدعي أنه أحدث معايير C.

<https://www.open-std.org/jtc1/sc22/wg14/www/docs/n1570.pdf>

R2 / مسودة عمل ++C رقم N4861

WG21، 2020. مسودة عامة من حقبة ++C، تشمل basic.fundamental، expr، numeric و cmath وعقود المكتبة. وتتعمد الأمثلة اتخاذ ++C أساساً.

<https://www.open-std.org/jtc1/sc22/wg21/docs/papers/2020/n4861.pdf>

R3 / دليل NIST/SEMATECH الإلكتروني: حدود الثقة للمتوسط

القسم 1.3.5.2. فترات المتوسط، وافتراضات أخذ العينات، ومعامل ستيودنت t.

<https://www.itl.nist.gov/div898/handbook/eda/section3/eda352.htm>

R4 / الحساب الدقيق لدالتي Log-Sum-Exp و Softmax

P. Blanchard, D. J. Higham, N. J. Higham. مجلة IMA Journal of Numerical Analysis، 2021: المسودة: arXiv:1909.03469. السلوك العددي للصيغ المزاحة.

<https://arxiv.org/abs/1909.03469>

R5 / التعلّم العميق (Deep Learning)

I. Goodfellow, Y. Bengio, A. Courville، دار MIT Press، 2016. تربط الفصول 2-6 الجبر الخطي والاحتمالات والحساب العددي والتعلّم.

<https://www.deeplearningbook.org/>

R6 / الانتباه هو كل ما تحتاجه (Attention Is All You Need)

A. Vaswani ومشاركون، 2017. انتباه الضرب النقطي المحجّم وبنية المحوّل (Transformer): arXiv:1706.03762.

<https://arxiv.org/abs/1706.03762>

R7 / تكميم الشبكات العصبية وتدريبها لاستدلال كفوّ بالحساب الصحيح وحده

B. Jacob ومشاركون، مؤتمر CVPR 2018: arXiv:1712.05877. التكميم التآلفي والاستدلال بالأعداد الصحيحة.

<https://arxiv.org/abs/1712.05877>

خريطة المصادر والموضوعات: R1-R2 للدروس 4-10 واستخدام المكتبة؛ R3 للدرس 15؛ R4 للدرس 27؛ R5 للدروس 17-22 و 26-28؛ R6 للدرس 29؛ R7 للدرس 30.

ما بَسْط، وما جرى التحقق منه

مسار أقصر عبر الطبعة الأولى

الأساسيات والموضوعات المتقطعة الأصلية. تجمع الدروس 1-6 و 9-12 الأفكار المتداخلة حول حسابات محددة. حساب الآلة والتحليل العددي الأصليان. تبقي الدروس 5-8 و 20-24 التمثيل والنطاق والخطأ بجوار الحساب. فصول الاحتمالات والأداء الأصلية. تربط الدروس 13-16 و 24-25 الافتراضات بقرارات القياس والسعة. الهندسة والتفاضل والتكامل ومسح التخصصات الأصلي. تعلّم الدروس 17-23 الجوهر المشترك؛ وتحفظ الصفحات 80-83 بجسور مركزة إلى الحقول المتخصصة. مادة البيانات والذكاء الاصطناعي الموسّعة. تضيف الدروس 26-30 والمشروعان C-D المعالجة المسبقة والخسائر والتدرجات والاسترجاع والانتباه والتكميم.

تصحيات وحدود أوضح

تجعل المراجعة فحوص الفيض صريحة قبل الضرب ونمو قوى الاثنين؛ وتميّز تساوي الطرفين التام من حساسية المسألة؛ وتحدد اصطلاحات المئينات؛ وتميّز أويلر الصريحة من شبه الضمنية؛ وتتجنب معاملة حلقة نيوتن ثابتة العدد بوصفها دالة عامة للجذر التربيعي.

سجل التحقق

بُنيت برامج الدروس الثلاثين وبرامج المشروعات الأربعة وشُغلت باستخدام GCC/G++ 13.3.0 على Linux x86-64، بمعيار C11 أو C++20 وخيارات التحسين والتحذيرات. وتتضمن المصادر فحوص الخرج وفحوصاً مخصصة للحدود والثوابت. وأُجريت فحوص إضافية للسلوك غير المعرّف على اختبارات الحدود العددية.

لم تُختبر Windows و macOS و MSVC والمعالجات الأخرى في تلك البيئة. تتجنب الأمثلة واجهات المنصات، لكن قد يختلف دعم المترجم والأرقام النهائية للفاصلة العائمة. وثبتت الاختبارات الحالات المفحوصة؛ ولا تبرهن على صحة عددية بلا قيود.

نطاق معلن بوضوح

هذه طبعة عملية للتعلّم. تقدم الرياضيات اللازمة للاستدلال على مكتبات وأنظمة أكبر؛ ولا تنفذ مكتبة BLAS صناعية، أو FFT صناعياً، أو نواة هندسية، أو مكتبة تشفير، أو إطار تعلّم متكاملًا.

ابن الثقة بتغيير شيء واحد

تفهم المعادلة حين تستطيع التنبؤ بما يحدث بعد تغيير افتراضاتها أو مدخلاتها.

حساب موثوق في C/C++

اقرأ الدروس 1-10. واطرح كل نوع ووحدة ونطاق مدخل ومسار فشل. وعدّل حساب حجم مخزن مؤقت بحيث يرفض حاصل ضرب يفيض.

تحليل البيانات والقياس

اقرأ الدروس 13-16 والمشروع A. اجمع مجموعة بيانات صغيرة، واختر اصطلاحاً للمئينات، وبلغ عن قيمة معتادة وعن الذيل معاً.

الهندسة والرسومات والمحاكاة

اقرأ الدروس 17-23 والمشروع B. ارسم الإطار الإحداثي، واختر قاعدة حدود، وقارن نتيجة عددية بحالة دقيقة صغيرة.

رياضيات عملية للذكاء الاصطناعي

اقرأ الدروس 17-21 و26-30، ثم المشروعين C-D. تحقق عددياً من تدرج واحد، وأبق المعالجة المسبقة مرتبطة ببيانات التدريب، واطرح معنى كل درجة.

هندسة الأداء

اقرأ الدروس 11-12 و24-25. توقع النمو، وعدّ البايتات، وحدّد الجزء المتوازي، وقس عدم اليقين قبل إعلان تسارع.

العادة التي تبقى

قبل الوثوق بنتيجة، أسأل: ماذا يعني هذا العدد؟ وما الافتراضات التي تجعله صالحاً؟ وما الحالة الصغيرة التي أستطيع فحصها باستقلال؟ وما الذي يتغير عند الحدود؟

أيمن الحراكي

رياضيات بلا خوف

فكرة تستطيع تصوّرُها. نتيجة تستطيع شرحها. شفرة تستطيع فحصها.

اجعل الرياضيات جزءاً عملياً من البرمجة: من حساب أمن للبايتات ومتوسط موثوق، إلى المتجهات والتعلّم والانتباه والتكثير.

30 درساً بصرياً

34 برنامجاً كاملاً بلغتي C و ++C

4 مشروعات مشروحة خطوة بخطوة

تمارين وشروح وأطلس موجز للمعادلات

إعداد أيمن الحراكي

الطبعة الثانية المنقّحة / أكتوبر 2026