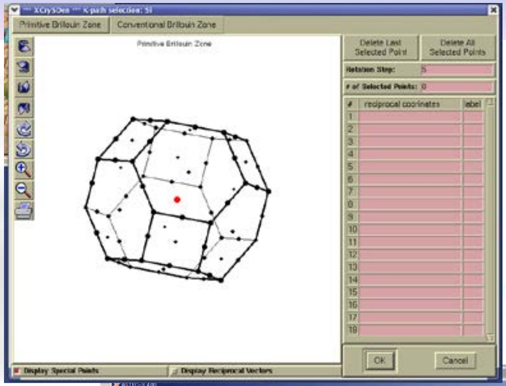
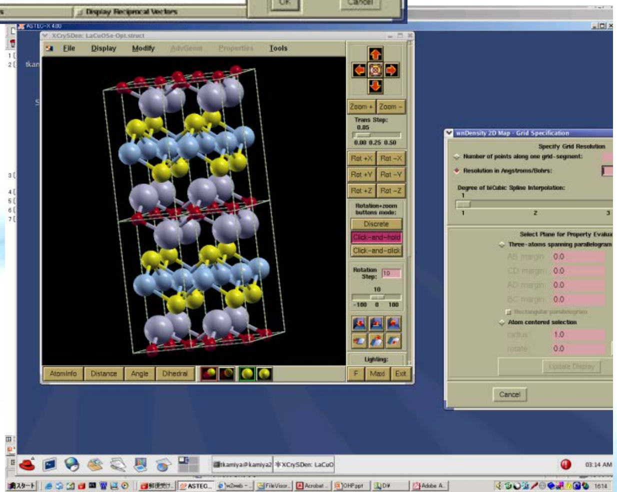
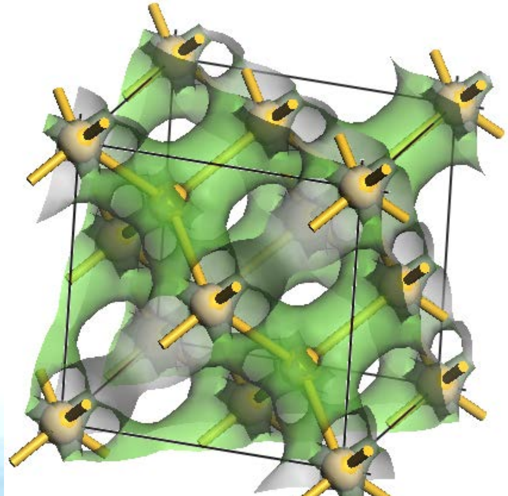
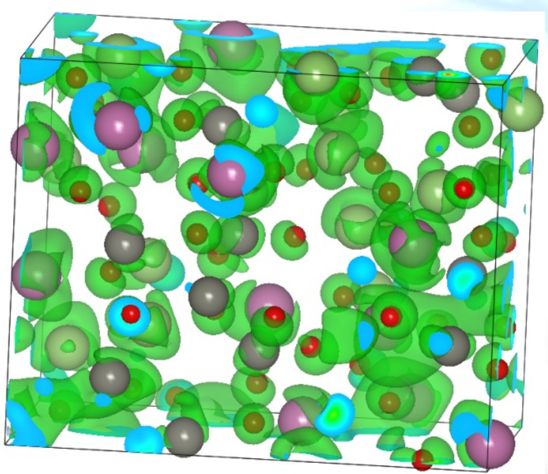
計算材料学特論

神谷利夫









講義予定

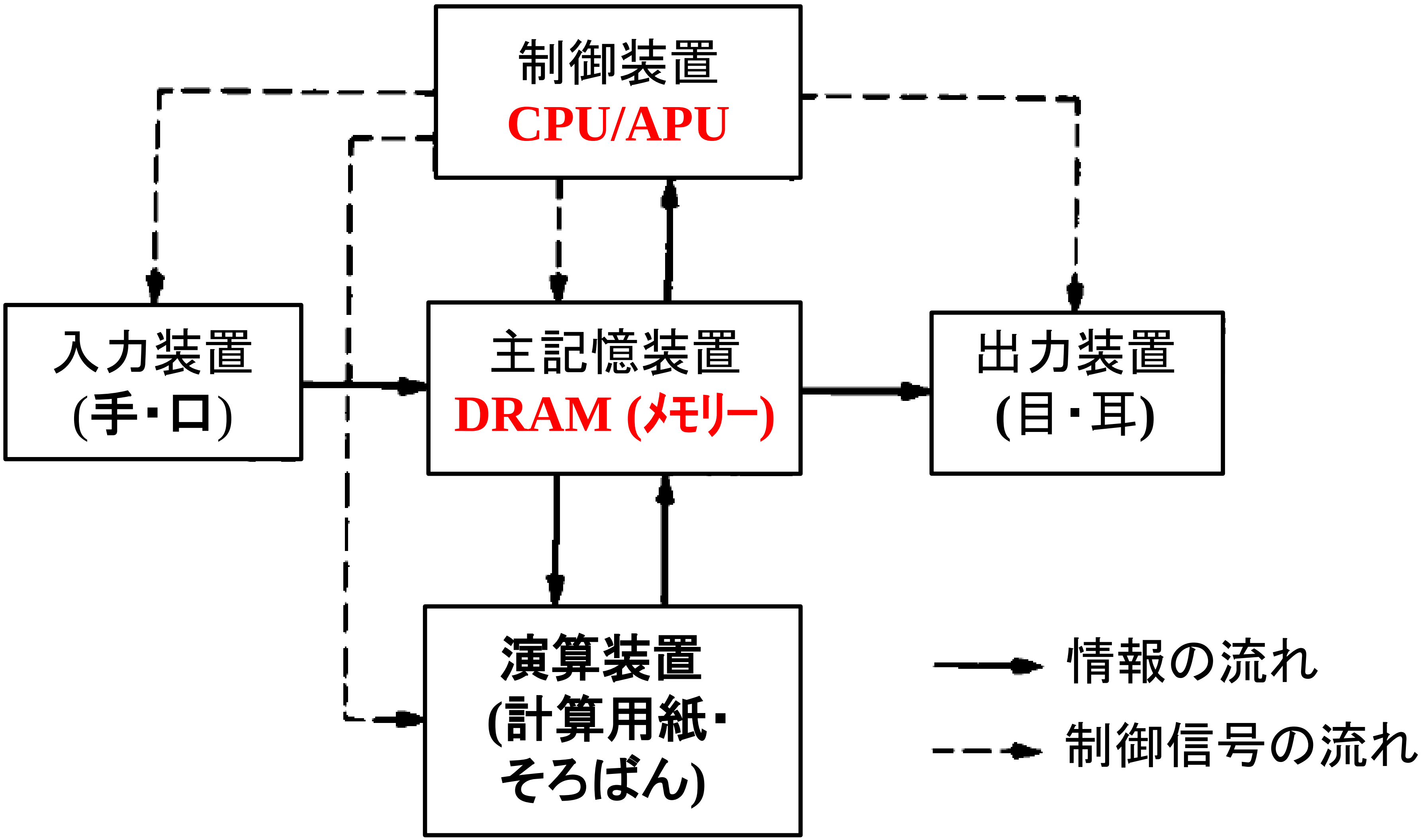
神谷担当講義資料配布: http://conf.msl.titech.ac.jp/Lecture/

6月12日	神谷1 (コンピュータの原理、数値微分・積分)
6月15日	神谷2 (数値積分、常微分方程式、分子動力学法)
6月19日	神谷3 (補間、平滑化、線形最小自乗、最適化)
6月22日	神谷4 (方程式の数値解、非線形最小自乗)
6月26日	神谷5 (Monte Carlo法、Fourier変換、行列)
6月29日	笹川1 (量子論おさらい1)
7月 3日	若井1 (微構造形成への応用、Monte Carlo法その１)
7月 6日	笹川2 (量子論おさらい2)
7月10日	若井2 (微構造形成への応用、Monte Carlo法その２)
7月13日	笹川3 (第一原理計算:基礎)
7月17日	若井3 (微構造形成への応用、Phase Field法)
7月20日	笹川4 (第一原理計算:応用1)
7月24日	若井4 (有限要素法その１)
7月27日	笹川5 (第一原理計算:応用2)
7月31日	若井5(有限要素法その２)
8月 3日	期末試験

コンピュータの仕組み

基本的な計算機の構成

大河内他、基礎 電子計算機、実教出版



数の表現

大河内他、基礎 電子計算機、実教出版

10進数 $1975 = 1\times1000 + 9\times100 + 7\times10 + 5\times1$

$= 1\times10^3 + 9\times10^2 + 7\times10^1 + 5\times10^0$

1000の位 100の位 10の位 1の位

2進数 $(11011)_2 = 1\times2^4 + 1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0$

$= 1\times(16)_{10} + 1\times(8)_{10} + 0\times(4)_{10} + 1\times(2)_{10} + 1\times(1)_{10}$

$= (27)_{10}$

r **進数** $N = a_n r^n + a_{n-1} r^{n-1} + \cdots\cdots + a_3 r^3 + a_2 r^2 + a_1 r^1 + a_0 r^0$

$= (a_n a_{n-1} \cdots\cdots a_3 a_2 a_1 a_0)_r$

r**: 基数**

数の表現

大河内他、基礎 電子計算機、実教出版

8進数　(01234567)

2桁: $0 \sim 8^2 - 1 = 63$

00: $0\times8^1 + 0\times8^0 = 0$

53: $5\times8^1 + 3\times8^0 = 43$

77: $7\times8^1 + 7\times8^0 = 63$

16進数 (0123456789ABCDEF) = (0 ~ 15)

2桁: $0 \sim 16^2-1=255$

00: $0\times16^1 + 0\times16^0 = 0$

9F: $9\times16^1 + 15\times16^0 = 159$

FF: $15\times16^1 + 15\times16^0 = 255$

64進数 (Base64)

(ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
0123456789+/)　=　(0 ~ 63)

対応関係

大河内他、基礎 電子計算機、実教出版

10 進 法	2 進 法 (16)(8)(4)(2)(1)	8 進 法	16 進 法
0	0000	00	0
1	0001	01	1
2	0010	02	2
3	0011	03	3
4	0100	04	4
5	0101	05	5
6	0110	06	6
7	0111	07	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10

コンピュータ内のデータ単位

bit (b): binary: 0 or 1

現在の計算機では 8 bitをまとめて扱うのが基本

byte (B): $0 \sim 2^8 - 1 = 255$

1 kB = 2^{10} B = 1024 B

1 MB = 1024 kB = 1,048,576 B

1 TB = 1024 GB = 1024^2 MB = 1024^3 kB = 1024^4 B

ビット演算

論理否定 (ビット反転)

NOT 0 = 1; NOT 1 = 0

論理積

0 AND 0 = 0; 1 AND 0 = 0
0 AND 1 = 0; 1 AND 1 = 1

論理和

0 OR 0 = 0; 1 OR 0 = 1
0 OR 1 = 1; 1 OR 1 = 1

排他的論理和

0 XOR 0 = 0; 1 XOR 0 = 1
0 XOR 1 = 1; 1 XOR 1 = 0

コンピュータアーキテクチャ

8bit CPU: 8008
16bit CPU: 8086
32bit CPU: 80386, 80486, Pentium,,,
64bit CPU: Pentium Pro(?), Itanium, Core i,,,

Pentium Pro:
プロセッサ 32bit: CPU内での命令･データ処理単位
外部データバス 64bit: メモリー･外部記憶装置とのデータ送受単位
浮動小数点 80bit: 実数を扱うデータ単位

数の表現: 整数型

整数型: CPUのプロセッサbit数が基本

16bit CPUなら16bit

符号無し整数型　0 ~ 2^{16}-1 = 65,535

符号付き整数型　-32,768 ~ +32,767

C言語: CPUのプロセッサbit数が基本

符号無し整数型 (32bit CPU)

unsigned int (int)　　0 ~ 4,294,967,295

符号付き整数型(32bit CPU)

signed int (int)　　-2,147,483,648 ~ + 2,147,483,647

short int　　: 16 bit

long int　　: 32 bit

long long int: 64 bit

数の表現: 浮動小数点型 (実数)

浮動小数点型: 最低 32bit

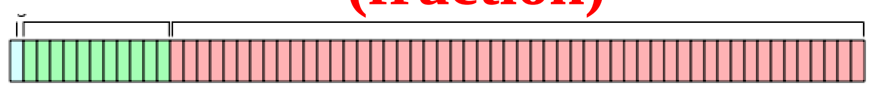
表現範囲はコンピュータアーキテクチャ、プログラム言語によって変わる

$$-(1.334343)_2 \times 2^{-015}$$

符号(sign)

仮数部 (fraction)

指数部 (exponent)



C言語

float : 32 bit 3.4E-38 ~ 3.4E+38

double : 64 bit 1.7E-308 ~ 1.7E+308

long double: 64 bit

Fortran

単精度浮動小数点 : 32 bit

倍精度浮動小数点 : 64 bit, 10進数で16桁

4倍精度浮動小数点: 128 bit

IEEE 754定義:

符号ビット: 1 bit

指数部 : 8 bit (単精度, -128 ~ +127) 11 bit (倍精度, -1024 ~ +1023)

仮数部 : 23 bit (単精度) 52bit (倍精度)

8,388,608: 7桁 4,503,599,627,370,495: 16桁

必要な整数型長: 文字の表現

英数字文字:

0~9, A~Z, a~z, その他 印字記号

非印字文字(制御文字)

ASCIIコード: 7 bit (0 ~ 127)

拡張ASCII文字コード

非英語文字、記号など: 8 bit

日本語

ASCII+半角カナ: 8 bit

漢字・かな (全角文字): 16 bit

現在: 全世界共通文字コード

Unicode: 最初は 2 Byte

現在 UCS-4で4 Byte

制御文字	10進	16進	文字	コード	10進	16進	文字	10進	16進	文字	10進	16進	文字
^@	0	00		NUL	32	20		64	40	@	96	60	`
^A	1	01		SOH	33	21	!	65	41	A	97	61	a
^B	2	02		STX	34	22	"	66	42	B	98	62	b
^C	3	03		ETX	35	23	#	67	43	C	99	63	c
^D	4	04		EOT	36	24	$	68	44	D	100	64	d
^E	5	05		ENQ	37	25	%	69	45	E	101	65	e
^F	6	06		ACK	38	26	&	70	46	F	102	66	f
^G	7	07		BEL	39	27	'	71	47	G	103	67	g
^H	8	08		BS	40	28	(	72	48	H	104	68	h
^I	9	09		HT	41	29	)	73	49	I	105	69	i
^J	10	0A		LF	42	2A	*	74	4A	J	106	6A	j
^K	11	0B		VT	43	2B	+	75	4B	K	107	6B	k
^L	12	0C		FF	44	2C	,	76	4C	L	108	6C	l
^M	13	0D		CR	45	2D	-	77	4D	M	109	6D	m
^N	14	0E		SO	46	2E	.	78	4E	N	110	6E	n
^O	15	0F		SI	47	2F	/	79	4F	O	111	6F	o
^P	16	10		DLE	48	30	0	80	50	P	112	70	p
^Q	17	11		DC1	49	31	1	81	51	Q	113	71	q
^R	18	12		DC2	50	32	2	82	52	R	114	72	r
^S	19	13		DC3	51	33	3	83	53	S	115	73	s
^T	20	14		DC4	52	34	4	84	54	T	116	74	t
^U	21	15		NAK	53	35	5	85	55	U	117	75	u
^V	22	16		SYN	54	36	6	86	56	V	118	76	v
^W	23	17		ETB	55	37	7	87	57	W	119	77	w
^X	24	18		CAN	56	38	8	88	58	X	120	78	x
^Y	25	19		EM	57	39	9	89	59	Y	121	79	y
^Z	26	1A		SUB	58	3A	:	90	5A	Z	122	7A	z
^[	27	1B		ESC	59	3B	;	91	5B	[	123	7B	{
^\	28	1C		FS	60	3C	<	92	5C	\	124	7C	|
^]	29	1D		GS	61	3D	=	93	5D	]	125	7D	}
^^	30	1E	▲	RS	62	3E	>	94	5E	^	126	7E	~
^_	31	1F	▼	US	63	3F	?	95	5F	_	127	7F	⌂*

必要な整数型長

unsigined int (16 bit): 65,536
　アドレスバスが16bitだと、メモリーで 64 kB

unsigined int (32 bit): 4,294,967,295
　アドレスバスが32bitだと、メモリーで 4 GB

日本のGDP: 約500兆円 = 500,000,000,000,000
(16桁必要)

unsigned long long int (64 bit): ~1.8E+19 (18桁)

円周率: 2013年時点 12.1兆桁
　多倍長計算: ソフトウェアで実現

必要な浮動小数点型長: 量子計算

原子の 1s 軌道エネルギー
H原子: 13.6 eV
重原子: >> keV

物性に関係するエネルギー
室温: 26 meV
磁性: 数 meV

物性を議論する量子計算では、meV ~ MeVの
9桁以上の計算精度が必要

IEEE 754定義:
仮数部　: 23 bit (単精度)　52bit　(倍精度)
8,388,608: 7桁　4,503,599,627,370,495: 16桁

必要な浮動小数点型長: 半導体

$\exp(-E_g / k_B T)$

$E_g = 1.1$ eV
$k_B T = \quad 0.026$ eV ($T = 300$ K) => $\exp(-42) \sim 10^{-19}$

$E_g = 4.0$ eV
$k_B T = \quad 0.026$ eV ($T = 300$ K) => $\exp(-154) \sim 10^{-67}$
$k_B T = 0.00026$ eV ($T = 3$ K) => $\exp(-15400) \sim 10^{-5141}$

倍精度浮動小数点 ： 64 bit, 10進数で16桁, 指数部 -1024 ~ +1023
$2^{-1024} \sim 10^{-308}$

4倍精度浮動小数点: 128 bit
8倍精度浮動小数点: 256 bit

コンピュータの計算精度・誤差

データ長: 原理的な有効桁数の上限を決める

・積み残し誤差 (桁あふれ, overflow):
有効桁数を超える整数の和・積
絶対精度が必要なければ、浮動小数点を使う
(指数部の桁あふれに注意)
⇔ underflow
overflow、underflowとも、CPUやソフトウェアで検出可能
・桁落ち誤差　: 非常に近い数の差
例: 5√41 – 32 ~ 5*6.403 – 32 = 32.015 – 32 = 0.015
与えられた有効数字が4桁なのに、最終的には2桁に減る
近い数同士の引き算は避ける
・情報落ち　　: 絶対値に大差のある2数の加減算
例: 1000 + 1.456 = 1001 (有効数字4桁の場合)
・丸め誤差
浮動小数点では有効桁数以下の数値は誤差になる

計算過程における誤差

- 積み残し誤差 (桁あふれ, overflow)
- 桁落ち誤差 (微小数の多数回の和など)
- 丸め誤差
- **情報埋没**
- **打切り誤差**

  徐々に減衰する数値を無限に和を取る

  - **テーラー展開**
  - **Coulombエネルギーの和**

  計算時間と必要な精度に応じて、どこかで計算を打ち切る

- **収束精度・収束誤差**

  繰り返し計算において、どの精度で収束判定を行うか

- **物理モデルの誤差**

桁落ち誤差

微小数 h の多数回 N の和

和の蓄積

```
double x = 0.0;
for(int i = 0 ; i < N ; i++) {
        x = x + h;
}
```

hを足すごとに誤差が蓄積される

積で計算

```
double x = 0.0;
for(int i = 0 ; i < N ; i++) {
        x = x0 + i * h;
}
```

乗算が入るので
計算時間は不利だが
一回の計算の誤差だけ

h = 0.1

N	x += h	i*h	誤差
0	0.0000000000000	0.0000000000000	0
50	5.0000000000000	5.0000000000000	−1.78E−15
100	10.0000000000000	10.0000000000000	−1.95E−14
150	15.0000000000000	15.0000000000000	−3.73E−14
200	20.0000000000000	20.0000000000000	1.42E−14
250	25.0000000000001	25.0000000000000	8.53E−14
300	30.0000000000002	30.0000000000000	1.56E−13
350	35.0000000000002	35.0000000000000	2.27E−13
400	40.0000000000003	40.0000000000000	2.98E−13
450	45.0000000000004	45.0000000000000	3.69E−13
500	50.0000000000004	50.0000000000000	4.41E−13
550	55.0000000000005	55.0000000000000	5.12E−13
600	60.0000000000006	60.0000000000000	5.83E−13
650	65.0000000000006	65.0000000000000	5.83E−13
700	70.0000000000003	70.0000000000000	2.98E−13
750	75.0000000000000	75.0000000000000	1.42E−14
800	79.9999999999997	80.0000000000000	−2.70E−13
850	84.9999999999994	85.0000000000000	−5.54E−13
900	89.9999999999992	90.0000000000000	−8.38E−13
950	94.9999999999989	95.0000000000000	−1.12E−12
1000	99.9999999999986	100.0000000000000	−1.41E−12

浮動小数点型 (実数) の誤差

浮動小数点型のコンピュータ内の表現:

$-(1.334343)_2 \times 2^{-015}$

- **誤差がない数値: 仮数部が有効数字内で 2^n の場合**

$1.0 = (1.0)_2 \times 2^0$

$0.5 = (1.0)_2 \times 2^{-1}$

$0.125 = (1.0)_2 \times 2^{-3}$

$0.039063 = 1.25 \times 2^{-5} = (1.01)_2 \times 2^{-5}$

$100.0 = 1.5625 \times 64 = (1+2^{-1}+2^{-4}) \times 2^6 = (1.1001)_2 \times 2^4$

- **誤差がでる数値: 仮数部が有効数字内で 2^n の場合**

$0.1 = (1.1001100110011001\cdots)_2 \times 2^{-3}$

情報埋没

$$\exp(x) = \sum_{n=0} \frac{x^n}{n!}$$

exp(-40)を計算してみる

正負が交番する大きな数の和を取るために誤差が大きくなる

0:	1
1:	-39
2:	761
18:	7362017438356
19:	-15234692665065.7
20:	29958727541777.7
21:	-56123977614114.5
22:	100390031760235
23:	-171808245412547: 正負の大数を交互に加えている
106:	3.03783540162816
107:	2.82317874249991
137:	2.88129986210968: 収束しているが、18桁の誤差
150:	2.88129986210968
(精確値	$4.2483542552915\times10^{-18}$**)**