

Tema 7: Regiones

1. Introducción
2. Crecimiento de Regiones
3. División y fusión

Bibliografía:

RM Haralick, LG Shapiro: **Computer and Robot Vision**, Addison-Wesley, 1992



1. Segmentación de Regiones

- Objetivo: dividir una imagen I en n regiones R_i *conexas* y *homogéneas* de acuerdo con un criterio $H(R_i)$, tales que:

$$1. I = \bigcup_{i=1}^n R_i$$

$$2. R_i \cap R_j = \emptyset \quad \forall i \neq j$$

$$3. H(R_i) = true \quad \forall i$$

$$4. H(R_i \cup R_j) = false \quad \forall R_i, R_j \text{ vecinos}, i \neq j$$

- El criterio $H(R_i)$ mide la homogeneidad de una cierta propiedad $P(\text{pixel})$, que depende de la aplicación:
 - Nivel de gris, color, textura,...
- La forma de medir $H(R_i)$ también depende de la aplicación



Criterios de Homogeneidad

- Distancia a un valor de referencia:
 - $|P(q) - \text{valor_ref}| < \text{umbral}$
- Distancia al valor medio:
 - $|P(q) - \text{media}_p(R_i)| < \text{umbral}$
- Desviación típica pequeña:
 - $\sigma_p(R_i) < \text{umbral}$
- Test de Hipótesis:
 - Puede ser que los valores de $P(q)$ encontrados en la región provengan de una distribución Gaussiana $N(m_p, \sigma_p)$?



2. Crecimiento de Regiones

$$1. I = \bigcup_{i=1}^n R_i$$

$$2. R_i \cap R_j = \emptyset \quad \forall i \neq j$$

$$3. H(R_i) = \text{true} \quad \forall i$$

$$4. H(R_i \cup R_j) = \text{false} \quad \forall R_i, R_j \text{ vecinos}, i \neq j$$

- Agrupar pixels similares o regiones en regiones mayores
- Crecer las regiones añadiendo pixels
 - Comenzar con un conjunto de pixels semilla (cumplen prop. 3)
 - Añadir vecinos a cada región mientras se cumpla 3
 - Cuando no se pueden añadir más, se cumple 4
 - #regiones = #semillas + 1 (fondo)
- Ejemplo: seguimiento de contornos con umbral con histéreseis
 - Semillas: magnitud > umbral_sup
 - Criterio: magnitud > umbral_inf



Crecimiento de Regiones

- Ejemplo: $P(i,j) = f(i,j)$ (intensidad)
 - Criterio $H(R)$: $|f(i,j) - f(\text{semilla})| < 3$

0	0	5	6	7
1	1	5	8	7
0	1	6	7	7
2	0	7	6	6
0	1	5	6	5

semillas

0	0	5	6	7
1	1	5	8	7
0	1	6	7	7
2	0	7	6	6
0	1	5	6	5

resultado

- Criterio $H(R)$: $|f(i,j) - f(\text{semilla})| < 8$

0	0	5	6	7
1	1	5	8	7
0	1	6	7	7
2	0	7	6	6
0	1	5	6	5

semillas

0	0	5	6	7
1	1	5	8	7
0	1	6	7	7
2	0	7	6	6
0	1	5	6	5

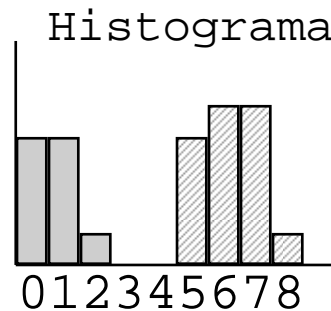
resultado



Elección de Semillas

- Usando información a priori del tipo de regiones que nos interesan:
 - Contorno: magnitud de gradiente elevada
 - Encontrar naranjas en un árbol: pixels de color naranja
 - Encontrar la carretera: pixels grises en la zona inferior image
- Calcular propiedades de los pixels y buscar grupos:
semillas = centro de los grupos

0	0	5	6	7
1	1	5	8	7
0	1	6	7	7
2	0	7	6	6
0	1	5	6	5



Semillas: 1 y 6



Test de Hipótesis para Fusión

- Supongamos que la propiedad $P(\cdot)$ sigue en cada región una distribución Gaussiana: $N(\mu_i, \sigma_i^2)$
- Dadas dos regiones vecinas R_1 y R_2 , con n_1 y n_2 pixels, Hay dos hipótesis alternativas:

H_0 : Los valores de $P(\cdot)$ pueden venir de la misma Gaussiana, con: $\mu_1 = \mu_2$ and $\sigma_1 = \sigma_2$.

Entonces, fusionar las regiones

H_1 : Los valores de $P(\cdot)$ vienen de dos Gaussianas diferentes
Entonces, no se pueden fusionar



Test de Hipótesis para Fusión

$$m_k = \frac{1}{n_k} \sum_{q \in R_k} P(q)$$

$$S_k^2 = \frac{1}{n_k - 1} \sum_{q \in R_k} [P(q) - m_k]^2$$

$$t = \frac{m_1 - m_2}{S_c \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

Sigue una distribución t de Student con $n_1 + n_2 - 2$ grados de libertad

$$S_c^2 = \frac{(n_1 - 1)S_1^2 + (n_2 - 1)S_2^2}{(n_1 + n_2 - 2)}$$

Estimación de la varianza conjunta

- Rechazar H_0 (no fusionar) si: $|t| > t_{1-\alpha/2, (n_1+n_2-2)}$
- Nivel de significación del test: α

Probabilidad de rechazar H_0 , siendo cierta

Probabilidad de no fusionar, siendo posible

Valores típicos: $\alpha = 0.01 - 0.05$



Test para añadir un pixel

Caso especial del test para añadir un pixel q ($n_2=1$) a una región R_k con n_k pixels:

$$m_k = \frac{1}{n_k} \sum_{r \in R_k} P(r)$$

$$S_k^2 = \frac{1}{n_k - 1} \sum_{r \in R_k} [P(r) - m_k]^2$$

$$t = \frac{P(q) - m_k}{S_k \sqrt{\frac{n_k + 1}{n_k}}}$$

Sigue una distribución t de Student con $n_k - 1$ grados de libertad

- Añadir el pixel si: $|t| \leq t_{1-\alpha/2, n_k-1}$
- Si se añade, actualizar la media y la varianza



Test para añadir un pixel

- Nivel de significación: α
Probabilidad de rechazar un pixel bueno
Valores típicos: $\alpha = 0.01 - 0.05$
- Ejemplo, para $\alpha = 0.05$
 - $n_k = 10$, $t_{0.975, 9} = 2.262$
 - $n_k = 50$, $t_{0.975, 49} = 2.009$
- Cuando se empieza una región ($n_k = 1$), usar un umbral a priori para $|P(q) - m_k|$



Distribución T de Student

$$\alpha = 0.05$$

df	75%	80%	85%	90%	95%	97.5%	99%	99.5%	99.75%	99.9%	99.95%
1	1.000	1.376	1.963	3.078	6.314	12.71	31.82	63.66	127.3	318.3	636.6
2	0.816	1.061	1.386	1.886	2.920	4.303	6.965	9.925	14.09	22.33	31.60
3	0.765	0.978	1.250	1.638	2.353	3.182	4.541	5.841	7.453	10.21	12.92
4	0.741	0.941	1.190	1.533	2.132	2.776	3.747	4.604	5.598	7.173	8.610
5	0.727	0.920	1.156	1.476	2.015	2.571	3.365	4.032	4.773	5.893	6.869
6	0.718	0.906	1.134	1.440	1.943	2.447	3.143	3.707	4.317	5.208	5.959
7	0.711	0.896	1.119	1.415	1.895	2.365	2.998	3.499	4.029	4.785	5.408
8	0.706	0.889	1.108	1.397	1.860	2.306	2.896	3.355	3.833	4.501	5.041
9	0.703	0.883	1.100	1.383	1.833	2.262	2.821	3.250	3.690	4.297	4.781
10	0.700	0.879	1.093	1.372	1.812	2.228	2.764	3.169	3.581	4.144	4.587
11	0.697	0.876	1.088	1.363	1.796	2.201	2.718	3.106	3.497	4.025	4.437
12	0.695	0.873	1.083	1.356	1.782	2.179	2.681	3.055	3.428	3.930	4.318
13	0.694	0.870	1.079	1.350	1.771	2.160	2.650	3.012	3.372	3.852	4.221
14	0.692	0.868	1.076	1.345	1.761	2.145	2.624	2.977	3.326	3.787	4.140
15	0.691	0.866	1.074	1.341	1.753	2.131	2.602	2.947	3.286	3.733	4.073
16	0.690	0.865	1.071	1.337	1.746	2.120	2.583	2.921	3.252	3.686	4.015
17	0.689	0.863	1.069	1.333	1.740	2.110	2.567	2.898	3.222	3.646	3.965
18	0.688	0.862	1.067	1.330	1.734	2.101	2.552	2.878	3.197	3.610	3.922
19	0.688	0.861	1.066	1.328	1.729	2.093	2.539	2.861	3.174	3.579	3.883
20	0.687	0.860	1.064	1.325	1.725	2.086	2.528	2.845	3.153	3.552	3.850
21	0.686	0.859	1.063	1.323	1.721	2.080	2.518	2.831	3.135	3.527	3.819
22	0.686	0.858	1.061	1.321	1.717	2.074	2.508	2.819	3.119	3.505	3.792
23	0.685	0.858	1.060	1.319	1.714	2.069	2.500	2.807	3.104	3.485	3.767
24	0.685	0.857	1.059	1.318	1.711	2.064	2.492	2.797	3.091	3.467	3.745
25	0.684	0.856	1.058	1.316	1.708	2.060	2.485	2.787	3.078	3.450	3.725
26	0.684	0.856	1.058	1.315	1.706	2.056	2.479	2.779	3.067	3.435	3.707
27	0.684	0.855	1.057	1.314	1.703	2.052	2.473	2.771	3.057	3.421	3.690
28	0.683	0.855	1.056	1.313	1.701	2.048	2.467	2.763	3.047	3.408	3.674
29	0.683	0.854	1.055	1.311	1.699	2.045	2.462	2.756	3.038	3.396	3.659
30	0.683	0.854	1.055	1.310	1.697	2.042	2.457	2.750	3.030	3.385	3.646
40	0.681	0.851	1.050	1.303	1.684	2.021	2.423	2.704	2.971	3.307	3.551
50	0.679	0.849	1.047	1.299	1.676	2.009	2.403	2.678	2.937	3.261	3.496
60	0.679	0.848	1.045	1.296	1.671	2.000	2.390	2.660	2.915	3.232	3.460
80	0.678	0.846	1.043	1.292	1.664	1.990	2.374	2.639	2.887	3.195	3.416
100	0.677	0.845	1.042	1.290	1.660	1.984	2.364	2.626	2.871	3.174	3.390
120	0.677	0.845	1.041	1.289	1.658	1.980	2.358	2.617	2.860	3.160	3.373
∞	0.674	0.842	1.036	1.282	1.645	1.960	2.326	2.576	2.807	3.090	3.291



Crecimiento Recursivo de Regiones

```
procedure recursive_growing
  choose_seeds
  for p in seeds
    create_region(R)
    add_pixel(p,R)
  endfor
end recursive_growing

procedure add_pixel(p,R)
  label(p,R)
  for q in neighbours(p)
    if not_labelled(q) and test_criterion(q,R) then
      add_pixel(q,R)
    endif
  endfor
end add_pixel
```

- Eligiendo semillas apropiadas, el algoritmo trabaja en las regiones interesantes.
- Algunos pixels quedarán sin etiquetar (fondo)



Crecimiento en una Pasada

```
procedure one_pass_growing
  for i:= 1 to rows
    for j:= 1 to cols
      Rleft:= Region(i,j-1)
      Cleft:= test_criterion(i,j,Rleft)
      Rsup:= Region(i-1,j)
      Csup:= test_criterion(i,j,Rsup)
      (C,R):=min(Cleft,Rleft,Csup,Rsup)
      if Cleft<=th and Csup<=th
        and test_fusion(Rleft,Rsup) then
          R=fusion(Rleft,Rsup)
        endif
      if C<=th then
        add_pixel(i,j,R)
      else
        create_region(i,j)
      endif
    endfor
  endfor
end one_pass_growing
```

Si no se añade, crear nueva región

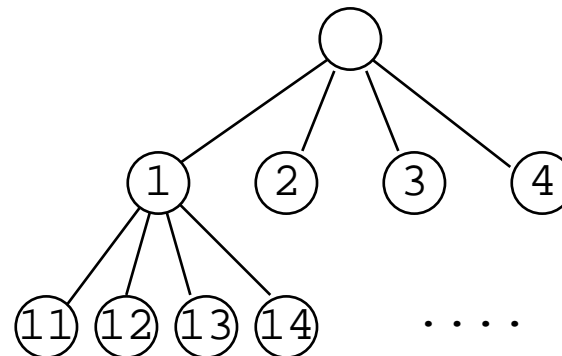
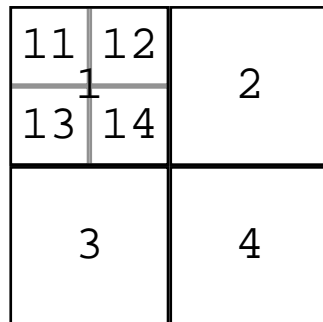
- » Etiqueta la imagen completa
- » Pueden aparecer regiones sin interés



3. División y Fusión

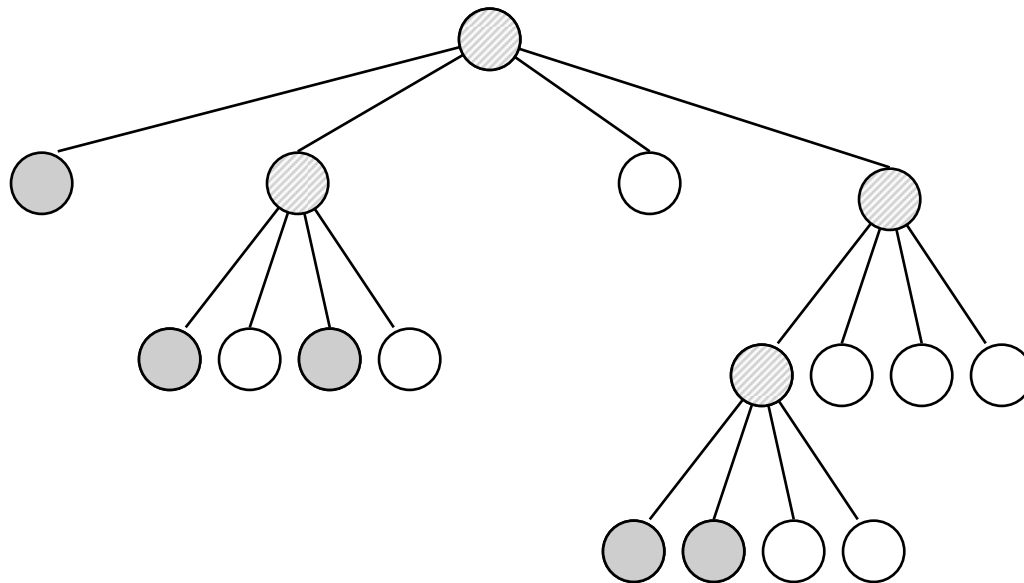
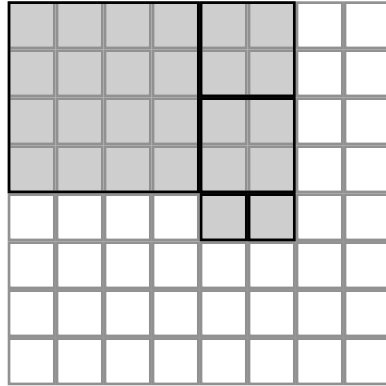
1. $I = \bigcup_{i=1}^n R_i$
2. $R_i \cap R_j = \emptyset \quad \forall i \neq j$
3. $H(R_i) = true \quad \forall i$
4. $H(R_i \cup R_j) = false \quad \forall R_i, R_j \text{ vecinos}, i \neq j$

- Empezar con la imagen completa
 - Partir la región hasta que se cumpla 3
 - Fusionar regiones vecinas hasta que se cumpla 4
- Quad-tree:



División y Fusión

- Ejemplo de Quad-tree:



Test para partir en cuatro

Hypothesis test:

- Supongamos que la propiedad $P(\cdot)$ sigue en cada región una distribución Gaussiana: $N(\mu_i, \sigma_i^2)$
- Dada una región R con $4n$ pixels , elegir entre dos hipótesis:

H_0 : Las medias y varianzas de $P(q)$ on las cuatro sub-regiones pueden ser la mismas

No partir la región

H_1 : Las medias y varianzas de $P(q)$ on las cuatro sub-regiones no pueden ser la mismas

Partir la región en cuatro



Test para partir en cuatro

$$m_k = \frac{1}{n} \sum_{p \in R_k} P(p) \quad k = 1..4$$

$$m = \frac{1}{4} \sum_{k=1}^4 m_k$$

$$f = \frac{n \sum_{k=1}^4 (m_k - m)^2 / 3}{\sum_{k=1}^4 \sum_{p \in R_k} [P(p) - m_k]^2 / 4(n-1)}$$

Sigue una distribución F de Fisher con 3 y 4(n-1) grados de libertad

- Rechazar H_0 (partir en cuatro) si: $|f| > f_{1-\alpha, 3, 4(n-1)}$
- Nivel de significación: α
Probabilidad de partir una región siendo innecesario
Valores típicos: $\alpha = 0.01 - 0.05$
- Ejemplo: región 16x16, $n = 64$, $\alpha = 0.05$
 $f_{0.95, 3, 252} = 2.6049$



Distribución F para $\alpha = 0.05$

df2/df1	1	2	3	4	5	6	7	8	9	10	12	15	20	24	30	40	60	120	INF
1	161.44	199.50	215.70	224.58	230.16	233.98	236.76	238.88	240.54	241.88	243.90	245.94	248.01	249.05	250.09	251.14	252.19	253.25	254.31
2	18.512	19.000	19.164	19.246	19.296	19.329	19.353	19.371	19.384	19.395	19.412	19.429	19.445	19.454	19.462	19.470	19.479	19.487	19.495
3	10.128	9.5521	9.2766	9.1172	9.0135	8.9406	8.8867	8.8452	8.8123	8.7855	8.7446	8.7029	8.6602	8.6385	8.6166	8.5944	8.5720	8.5494	8.5264
4	7.7086	6.9443	6.5914	6.3882	6.2561	6.1631	6.0942	6.0410	5.9988	5.9644	5.9117	5.8578	5.8025	5.7744	5.7459	5.7170	5.6877	5.6581	5.6281
5	6.6079	5.7861	5.4095	5.1922	5.0503	4.9503	4.8759	4.8183	4.7725	4.7351	4.6777	4.6188	4.5581	4.5272	4.4957	4.4638	4.4314	4.3985	4.3650
6	5.9874	5.1433	4.7571	4.5337	4.3874	4.2839	4.2067	4.1468	4.0990	4.0600	3.9999	3.9381	3.8742	3.8415	3.8082	3.7743	3.7398	3.7047	3.6689
7	5.5914	4.7374	4.3468	4.1203	3.9715	3.8660	3.7870	3.7257	3.6767	3.6365	3.5747	3.5107	3.4445	3.4105	3.3758	3.3404	3.3043	3.2674	3.2298
8	5.3177	4.4590	4.0662	3.8379	3.6875	3.5806	3.5005	3.4381	3.3881	3.3472	3.2839	3.2184	3.1503	3.1152	3.0794	3.0428	3.0053	2.9669	2.9276
9	5.1174	4.2565	3.8625	3.6331	3.4817	3.3738	3.2927	3.2296	3.1789	3.1373	3.0729	3.0061	2.9365	2.9005	2.8637	2.8259	2.7872	2.7475	2.7067
10	4.9646	4.1028	3.7083	3.4780	3.3258	3.2172	3.1355	3.0717	3.0204	2.9782	2.9130	2.8450	2.7740	2.7372	2.6996	2.6609	2.6211	2.5801	2.5379
11	4.8443	3.9823	3.5874	3.3567	3.2039	3.0946	3.0123	2.9480	2.8962	2.8536	2.7876	2.7186	2.6464	2.6090	2.5705	2.5309	2.4901	2.4480	2.4045
12	4.7472	3.8853	3.4903	3.2592	3.1059	2.9961	2.9134	2.8486	2.7964	2.7534	2.6866	2.6169	2.5436	2.5055	2.4663	2.4259	2.3842	2.3410	2.2962
13	4.6672	3.8056	3.4105	3.1791	3.0254	2.9153	2.8321	2.7669	2.7144	2.6710	2.6037	2.5331	2.4589	2.4202	2.3803	2.3392	2.2966	2.2524	2.2064
14	4.6001	3.7389	3.3439	3.1122	2.9582	2.8477	2.7642	2.6987	2.6458	2.6022	2.5342	2.4630	2.3879	2.3487	2.3082	2.2664	2.2229	2.1778	2.1307
15	4.5431	3.6823	3.2874	3.0556	2.9013	2.7905	2.7066	2.6408	2.5876	2.5437	2.4753	2.4034	2.3275	2.2878	2.2468	2.2043	2.1601	2.1141	2.0658
16	4.4940	3.6337	3.2389	3.0069	2.8524	2.7413	2.6572	2.5911	2.5377	2.4935	2.4247	2.3522	2.2756	2.2354	2.1938	2.1507	2.1058	2.0589	2.0096
17	4.4513	3.5915	3.1968	2.9647	2.8100	2.6987	2.6143	2.5480	2.4943	2.4499	2.3807	2.3077	2.2304	2.1898	2.1477	2.1040	2.0584	2.0107	1.9604
18	4.4139	3.5546	3.1599	2.9277	2.7729	2.6613	2.5767	2.5102	2.4563	2.4117	2.3421	2.2686	2.1906	2.1497	2.1071	2.0629	2.0166	1.9681	1.9168
19	4.3807	3.5219	3.1274	2.8951	2.7401	2.6283	2.5435	2.4768	2.4227	2.3779	2.3080	2.2341	2.1555	2.1141	2.0712	2.0264	1.9795	1.9302	1.8780
20	4.3512	3.4928	3.0984	2.8661	2.7109	2.5990	2.5140	2.4471	2.3928	2.3479	2.2776	2.2033	2.1242	2.0825	2.0391	1.9938	1.9464	1.8963	1.8432
21	4.3248	3.4668	3.0725	2.8401	2.6848	2.5727	2.4876	2.4205	2.3660	2.3210	2.2504	2.1757	2.0960	2.0540	2.0102	1.9645	1.9165	1.8657	1.8117
22	4.3009	3.4434	3.0491	2.8167	2.6613	2.5491	2.4638	2.3965	2.3419	2.2967	2.2258	2.1508	2.0707	2.0283	1.9842	1.9380	1.8894	1.8380	1.7831
23	4.2793	3.4221	3.0280	2.7955	2.6400	2.5277	2.4422	2.3748	2.3201	2.2747	2.2036	2.1282	2.0476	2.0050	1.9605	1.9139	1.8648	1.8128	1.7570
24	4.2597	3.4028	3.0088	2.7763	2.6207	2.5082	2.4226	2.3551	2.3002	2.2547	2.1834	2.1077	2.0267	1.9838	1.9390	1.8920	1.8424	1.7896	1.7330
25	4.2417	3.3852	2.9912	2.7587	2.6030	2.4904	2.4047	2.3371	2.2821	2.2365	2.1649	2.0889	2.0075	1.9643	1.9192	1.8718	1.8217	1.7684	1.7110
26	4.2252	3.3690	2.9752	2.7426	2.5868	2.4741	2.3883	2.3205	2.2655	2.2197	2.1479	2.0716	1.9898	1.9464	1.9010	1.8533	1.8027	1.7488	1.6906
27	4.2100	3.3541	2.9604	2.7278	2.5719	2.4591	2.3732	2.3053	2.2501	2.2043	2.1323	2.0558	1.9736	1.9299	1.8842	1.8361	1.7851	1.7306	1.6717
28	4.1960	3.3404	2.9467	2.7141	2.5581	2.4453	2.3593	2.2913	2.2360	2.1900	2.1179	2.0411	1.9586	1.9147	1.8687	1.8203	1.7689	1.7138	1.6541
29	4.1830	3.3277	2.9340	2.7014	2.5454	2.4324	2.3463	2.2783	2.2229	2.1768	2.1045	2.0275	1.9446	1.9005	1.8543	1.8055	1.7537	1.6981	1.6376
30	4.1709	3.3158	2.9223	2.6896	2.5336	2.4205	2.3343	2.2662	2.2107	2.1646	2.0921	2.0148	1.9317	1.8874	1.8409	1.7918	1.7396	1.6835	1.6223
40	4.0847	3.2317	2.8387	2.6060	2.4495	2.3359	2.2490	2.1802	2.1240	2.0772	2.0035	1.9245	1.8389	1.7929	1.7444	1.6928	1.6373	1.5766	1.5089
60	4.0012	3.1504	2.7581	2.5252	2.3683	2.2541	2.1665	2.0970	2.0401	1.9926	1.9174	1.8364	1.7480	1.7001	1.6491	1.5943	1.5343	1.4673	1.3893
120	3.9201	3.0718	2.6802	2.4472	2.2899	2.1750	2.0868	2.0164	1.9588	1.9105	1.8337	1.7505	1.6587	1.6084	1.5543	1.4952	1.4290	1.3519	1.2539
INF	3.8415	2.9957	2.6049	2.3719	2.2141	2.0986	2.0096	1.9384	1.8799	1.8307	1.7522	1.6664	1.5705	1.5173	1.4591	1.3940	1.3180	1.2214	1.0000



Distribución F para $\alpha = 0.01$

df2/df1	1	2	3	4	5	6	7	8	9	10	12	15	20	24	30	40	60	120	INF
1	4052.1	4999.5	5403.3	5624.5	5763.6	5858.9	5928.3	5981.0	6022.4	6055.8	6106.3	6157.2	6208.7	6234.6	6260.6	6286.7	6313.0	6339.3	6365.8
2	98.503	99.000	99.166	99.249	99.299	99.333	99.356	99.374	99.388	99.399	99.416	99.433	99.449	99.458	99.466	99.474	99.482	99.491	99.499
3	34.116	30.817	29.457	28.710	28.237	27.911	27.672	27.489	27.345	27.229	27.052	26.872	26.690	26.598	26.505	26.411	26.316	26.221	26.125
4	21.198	18.000	16.694	15.977	15.522	15.207	14.976	14.799	14.659	14.546	14.374	14.198	14.020	13.929	13.838	13.745	13.652	13.558	13.463
5	16.258	13.274	12.060	11.392	10.967	10.672	10.456	10.289	10.158	10.051	9.888	9.722	9.553	9.466	9.379	9.291	9.202	9.112	9.020
6	13.745	10.925	9.780	9.148	8.746	8.466	8.260	8.102	7.976	7.874	7.718	7.559	7.396	7.313	7.229	7.143	7.057	6.969	6.880
7	12.246	9.547	8.451	7.847	7.460	7.191	6.993	6.840	6.719	6.620	6.469	6.314	6.155	6.074	5.992	5.908	5.824	5.737	5.650
8	11.259	8.649	7.591	7.006	6.632	6.371	6.178	6.029	5.911	5.814	5.667	5.515	5.359	5.279	5.198	5.116	5.032	4.946	4.859
9	10.561	8.022	6.992	6.422	6.057	5.802	5.613	5.467	5.351	5.257	5.111	4.962	4.808	4.729	4.649	4.567	4.483	4.398	4.311
10	10.044	7.559	6.552	5.994	5.636	5.386	5.200	5.057	4.942	4.849	4.706	4.558	4.405	4.327	4.247	4.165	4.082	3.996	3.909
11	9.646	7.206	6.217	5.668	5.316	5.069	4.886	4.744	4.632	4.539	4.397	4.251	4.099	4.021	3.941	3.860	3.776	3.690	3.602
12	9.330	6.927	5.953	5.412	5.064	4.821	4.640	4.499	4.388	4.296	4.155	4.010	3.858	3.780	3.701	3.619	3.535	3.449	3.361
13	9.074	6.701	5.739	5.205	4.862	4.620	4.441	4.302	4.191	4.100	3.960	3.815	3.665	3.587	3.507	3.425	3.341	3.255	3.165
14	8.862	6.515	5.564	5.035	4.695	4.456	4.278	4.140	4.030	3.939	3.800	3.656	3.505	3.427	3.348	3.266	3.181	3.094	3.004
15	8.683	6.359	5.417	4.893	4.556	4.318	4.142	4.004	3.895	3.805	3.666	3.522	3.372	3.294	3.214	3.132	3.047	2.959	2.868
16	8.531	6.226	5.292	4.773	4.437	4.202	4.026	3.890	3.780	3.691	3.553	3.409	3.259	3.181	3.101	3.018	2.933	2.845	2.753
17	8.400	6.112	5.185	4.669	4.336	4.102	3.927	3.791	3.682	3.593	3.455	3.312	3.162	3.084	3.003	2.920	2.835	2.746	2.653
18	8.285	6.013	5.092	4.579	4.248	4.015	3.841	3.705	3.597	3.508	3.371	3.227	3.077	2.999	2.919	2.835	2.749	2.660	2.566
19	8.185	5.926	5.010	4.500	4.171	3.939	3.765	3.631	3.523	3.434	3.297	3.153	3.003	2.925	2.844	2.761	2.674	2.584	2.489
20	8.096	5.849	4.938	4.431	4.103	3.871	3.699	3.564	3.457	3.368	3.231	3.088	2.938	2.859	2.778	2.695	2.608	2.517	2.421
21	8.017	5.780	4.874	4.369	4.042	3.812	3.640	3.506	3.398	3.310	3.173	3.030	2.880	2.801	2.720	2.636	2.548	2.457	2.360
22	7.945	5.719	4.817	4.313	3.988	3.758	3.587	3.453	3.346	3.258	3.121	2.978	2.827	2.749	2.667	2.583	2.495	2.403	2.305
23	7.881	5.664	4.765	4.264	3.939	3.710	3.539	3.406	3.299	3.211	3.074	2.931	2.781	2.702	2.620	2.535	2.447	2.354	2.256
24	7.823	5.614	4.718	4.218	3.895	3.667	3.496	3.363	3.256	3.168	3.032	2.889	2.738	2.659	2.577	2.492	2.403	2.310	2.211
25	7.770	5.568	4.675	4.177	3.855	3.627	3.457	3.324	3.217	3.129	2.993	2.850	2.699	2.620	2.538	2.453	2.364	2.270	2.169
26	7.721	5.526	4.637	4.140	3.818	3.591	3.421	3.288	3.182	3.094	2.958	2.815	2.664	2.585	2.503	2.417	2.327	2.233	2.131
27	7.677	5.488	4.601	4.106	3.785	3.558	3.388	3.256	3.149	3.062	2.926	2.783	2.632	2.552	2.470	2.384	2.294	2.198	2.097
28	7.636	5.453	4.568	4.074	3.754	3.528	3.358	3.226	3.120	3.032	2.896	2.753	2.602	2.522	2.440	2.354	2.263	2.167	2.064
29	7.598	5.420	4.538	4.045	3.725	3.499	3.330	3.198	3.092	3.005	2.868	2.726	2.574	2.495	2.412	2.325	2.234	2.138	2.034
30	7.562	5.390	4.510	4.018	3.699	3.473	3.304	3.173	3.067	2.979	2.843	2.700	2.549	2.469	2.386	2.299	2.208	2.111	2.006
40	7.314	5.179	4.313	3.828	3.514	3.291	3.124	2.993	2.888	2.801	2.665	2.522	2.369	2.288	2.203	2.114	2.019	1.917	1.805
60	7.077	4.977	4.126	3.649	3.339	3.119	2.953	2.823	2.718	2.632	2.496	2.352	2.198	2.115	2.028	1.936	1.836	1.726	1.601
120	6.851	4.787	3.949	3.480	3.174	2.956	2.792	2.663	2.559	2.472	2.336	2.192	2.035	1.950	1.860	1.763	1.656	1.533	1.381
INF	6.635	4.605	3.782	3.319	3.017	2.802	2.639	2.511	2.407	2.321	2.185	2.039	1.878	1.791	1.696	1.592	1.473	1.325	1.000

